



***Facultad
de
Ciencias***

APLICACIÓN WEB PARA LA GESTIÓN DE INCIDENCIAS EN FÁBRICA

**(Web application for factory
incident management)**

**Trabajo de Fin de Grado
para acceder al**

GRADO EN INGENIERÍA INFORMÁTICA

Autor: Jairo Diego Cuesta

Director: Cristina Tirnauca

Septiembre – 2019

Resumen

El presente trabajo de fin de grado expone el planteamiento, desarrollo y despliegue de un sistema web de gestión de incidencias sobre la empresa Ampuero Grupo Industrial 10, dirigida a la producción de ventanas de aluminio y PVC (Polyvinyl Chloride).

A lo largo de los últimos años, Ampuero Grupo Industrial 10 ha experimentado un constante y extraordinario crecimiento, basado en la consolidación de clientes importantes y manifestado a través de un volumen de producción con un aumento potencial, soportado en la misma medida, por un personal creciente.

El incremento sobre los factores expuestos conlleva de forma natural que en un determinado espacio tiempo, el volumen de anomalías sufridas durante el ciclo de vida de la ventana en fábrica se vea aumentado. De esta forma, se torna primordial disponer de la capacidad de controlar y optimizar la resolución de dichas anomalías, denominadas “incidencias”, con el objetivo de mantener el estándar de calidad y nivel de servicio propios de la marca.

Mediante una metodología ágil de desarrollo *software*, basadas en la entrega rápida e incremental de funcionalidad con el objetivo de aportar valor de negocio aun en etapas tempranas del proyecto, y mas concretamente con el empleo de su variante “programación extrema”, denominada de dicha forma por llevar al extremo las prácticas propuestas por el desarrollo ágil, se construirá una aplicación web que otorgue al usuario la capacidad de recolectar, gestionar, y resolver las incidencias ocurridas durante la fabricación y manipulación del producto, conformando de esta forma la base fundamental para la constitución de un gestor de calidad completo.

A términos de implementación, la construcción de la aplicación web se basará principalmente en lenguaje de servidor PHP (Hypertext Preprocessor), mientras que emplearemos el sistema de gestión de base de datos Microsoft SQL (Structured Query Language) Server para dar soporte a las necesidades de información de la herramienta.

Palabras clave: Aplicación Web, Desarrollo Ágil, Programación Extrema, PHP, SQL Server, Gestor Incidencias

Abstract

The present end of degree project exposes the approach, development and deployment of an incident management web system for the company Ampuero Grupo Industrial 10, aimed at the production of aluminum and PVC (Polyvinyl Chloride) windows.

Over the last years, Ampuero Grupo Industrial 10 has experienced a constant and extraordinary growth, based on some significant clients consolidation and manifested through a potential increase on the production volume, supported at the same time by staff growing.

The increase of these factors naturally implies that in a given space of time, the volume of anomalies suffered during the factory life cycle of the window will be increased. In this way, it becomes essential to have the ability to control and optimize the resolution of these anomalies, called “incidents”, with the aim of maintaining the standard of quality and level of service of the brand.

Through an agile software development methodology, based on the rapid and incremental delivery of functionality with the objective of providing business value even in the early stages of the project, and more specifically with the use of its “extreme programming” variant, called this way for taking to the extreme the practices proposed by the agile development, we will build a web application that gives the user the ability to collect, manage, and resolve the incidents that occurred during the manufacture and handling of the product, thus forming the fundamental basis for the constitution of a complete quality management system.

In terms of implementation, web application construction will be mainly based on PHP (Hypertext Preprocessor), server language, while we will use the Microsoft SQL (Structured Query Language) Server database management system to support the information needs of the tool.

Keywords: Web Application, Agile Development, Extreme Programming, PHP, SQL Server, Incident Management System

Contenidos

1	INTRODUCCIÓN.....	- 6 -
1.1	AMPUERO GRUPO INDUSTRIAL 10	- 6 -
1.2	MOTIVACIONES.....	- 7 -
1.3	ESTRUCTURA DEL DOCUMENTO.....	- 8 -
2	PRESENTACIÓN DEL PROBLEMA.....	- 9 -
2.1	IDIOSINCRASIA	- 9 -
2.2	PUNTO DE PARTIDA	- 11 -
2.3	NECESIDAD DE CAMBIO.....	- 11 -
2.4	OBJETIVOS	- 12 -
3	METODOLOGÍA.....	- 13 -
3.1	DESARROLLO ÁGIL.....	- 13 -
3.2	PROGRAMACIÓN EXTREMA (XP)	- 15 -
4	TECNOLOGÍAS Y HERRAMIENTAS	- 18 -
4.1	HTML5 + CSS3 + JAVASCRIPT + AJAX.....	- 18 -
4.2	PHP + SWIFT MAILER	- 18 -
4.3	MICROSOFT SQL SERVER + TRANSACT-SQL	- 18 -
4.4	MICROSOFT OFFICE + MICROSOFT POWERBI.....	- 18 -
5	PROCESO SOFTWARE.....	- 19 -
5.1	ROLES	- 19 -
5.2	HISTORIAS DE USUARIO.....	- 20 -
5.3	PLANIFICACIÓN.....	- 26 -
5.4	LIBERACIONES.....	- 29 -
5.4.1	TAREAS DE PROGRAMACIÓN	- 29 -
5.4.2	PRUEBAS UNITARIAS.....	- 31 -
5.4.3	IMPLEMENTACIÓN.....	- 33 -
5.4.3.1	BASE DE DATOS (BD)	- 33 -
5.4.3.2	WEB	- 35 -
5.4.4	PRUEBAS DE ACEPTACIÓN	- 38 -
6	TRABAJO FUTURO.....	- 46 -
6.1	GARANTIZADOS.....	- 46 -
6.2	A DEBATE.....	- 47 -
7	CONCLUSIONES.....	- 48 -
7.1	HERRAMIENTA.....	- 48 -
7.2	ACADÉMICO/PERSONAL	- 49 -
8	BIBLIOGRAFÍA	- 50 -

Ilustraciones

ILUSTRACIÓN 1.1: AGI10 - CRECIMIENTO ANUAL RESPECTO A LOS 5 ÚLTIMOS AÑOS	- 6 -
ILUSTRACIÓN 2.1: DE PRESUPUESTO A INSTANCIA	- 10 -
ILUSTRACIÓN 2.2: COMPOSICIÓN LOTE-CICLO DE PRODUCCIÓN.....	- 10 -
ILUSTRACIÓN 2.3: FORMULARIO PARA EL REGISTRO DE INCIDENCIAS PRE-DESARROLLO.....	- 11 -
ILUSTRACIÓN 3.1: COMPARATIVA MODELO EN CASCADA - DESARROLLO INCREMENTAL	- 14 -
ILUSTRACIÓN 3.2: COMPARATIVA DESARROLLO INCREMENTAL - PROGRAMACIÓN EXTREMA	- 16 -
ILUSTRACIÓN 5.1: JERARQUÍA DE PLANIFICACIÓN EN XP	- 26 -
ILUSTRACIÓN 5.2: PLANIFICACIÓN LIBERACIÓN 1 - RECOLECCIÓN DE INCIDENCIAS.....	- 27 -
ILUSTRACIÓN 5.3: PLANIFICACIÓN LIBERACIÓN 2 - BÚSQUEDA DE INCIDENCIAS	- 28 -
ILUSTRACIÓN 5.4: PLANIFICACIÓN LIBERACIÓN 3 - UNIFICACIÓN + GESTIÓN DE INCIDENCIAS.....	- 29 -
ILUSTRACIÓN 5.5: DIAGRAMA BD CORRESPONDIENTE A LA TERCERA LIBERACIÓN.....	- 33 -
ILUSTRACIÓN 5.6: ESTRUCTURA DE FICHEROS WEB	- 36 -
ILUSTRACIÓN 5.7: ESQUEMA COMUNICACIÓN ASÍNCRONA EN NUESTRO ENTORNO DE TRABAJO	- 37 -
ILUSTRACIÓN 5.8: PRUEBA DE ACEPTACIÓN 1.1 – ORIGEN INEXISTENTE.....	- 39 -
ILUSTRACIÓN 5.9: PRUEBA DE ACEPTACIÓN 4.1 – FORMULARIO DE INTRODUCCIÓN.....	- 41 -
ILUSTRACIÓN 5.10: PRUEBA DE ACEPTACIÓN 5.1 – CORREO DE NOTIFICACIÓN	- 42 -
ILUSTRACIÓN 5.11: PRUEBA DE ACEPTACIÓN 6.1 – LISTADO DE INCIDENCIAS (COMPLETO)	- 43 -
ILUSTRACIÓN 5.12: PRUEBA DE ACEPTACIÓN 9.1 – LISTADO DE INCIDENCIAS (DETALLE)	- 43 -
ILUSTRACIÓN 5.13: PRUEBA DE ACEPTACIÓN 12.1 – EDICIÓN FECHA ESTIMADA DE RECEPCIÓN	- 44 -
ILUSTRACIÓN 5.14: PRUEBA DE ACEPTACIÓN 14.1 – COLUMNA “ESTADO”	- 45 -

1 Introducción

Como introducción, en este primer capítulo (1) presentaremos una breve fotografía temporal de la empresa en la que se ha desarrollado el proyecto (1.1), de forma que podamos enlazar dicho escenario de crecimiento con las principales motivaciones (1.2) que han originado e impulsado este trabajo de fin de grado. Además, proporcionaremos a grandes rasgos una descripción de cómo hemos distribuido el contenido del resto de este informe en base a sus correspondientes capítulos y secciones (1.3).

1.1 Ampuero Grupo Industrial 10

Nos gustaría comenzar este informe con una pequeña contextualización evolutiva de la empresa Ampuero Grupo Industrial 10, en adelante AGI10, con el objetivo de que el lector pueda constatar como dicha evolución repercute directamente sobre la necesidad de contar con un gestor de incidencias internas, que permita la recolección, análisis y resolución de cualquier anomalía ocurrida a lo largo del ciclo de vida de la ventana en fábrica. Para ello, nos hemos apoyado en dos indicadores: el volumen de producción y la evolución del número de personal en estos últimos 5 años, ambos representados porcentualmente sobre el total y acotados a los diferentes meses de agosto con el objetivo de obtener la tendencia de este 2019.

Evolución Anual AGI10 - Producción y Personal

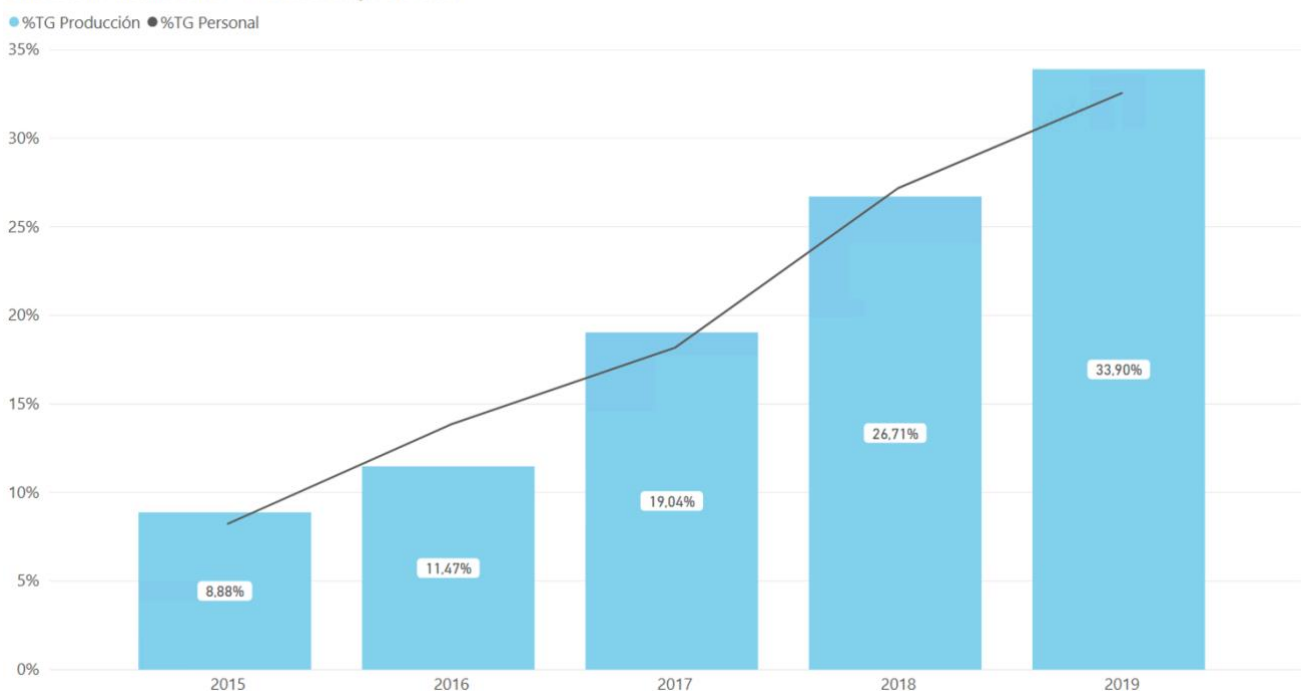


Ilustración 1.1: AGI10 - Crecimiento anual respecto a los 5 últimos años

1.2 Motivaciones

Una vez presentada la fotografía de progresión temporal, analicemos algunos de los principales factores afectados por el crecimiento experimentado, y cómo a partir de los mismos surgen las motivaciones para llevar a cabo este trabajo:

1. En primer lugar, un aumento potencial sobre el volumen de producción implicará en la misma medida un aumento sobre el número de situaciones de incidencia que se puedan dar en un determinado espacio temporal. Este incremento en la cantidad de incidencias sin un gestor apropiado, desembocaría no solo en una notable pérdida de control sobre el análisis y resolución de las mismas, sino, incluso, en una más que segura incapacidad para conocer su mera existencia.
2. Por otro lado, la necesidad de ampliar la capacidad de producción de cara a hacer frente al incremento en la entrada de ventas, ha permitido la introducción de un nuevo turno de trabajo, pasando de dos turnos, a una jornada ininterrumpida de veinticuatro horas distribuida en tres turnos de producción. A términos prácticos, esto provoca que muchas de las incidencias surgidas a lo largo del ciclo de vida de la ventana sean resueltas total o parcialmente por empleados que no estuvieron involucrados en la aparición o detección de las mismas. Un gestor de incidencias facilita este traspaso de información, haciendo dispensable una comunicación explícita para acometer una resolución. Mediante este sistema un empleado podría conocer de forma asequible el estado de una determinada incidencia sobre la que se le considere responsable, y tomar una decisión resolutive basándose en la información registrada.
3. En contraposición con la tónica mantenida hasta el momento, existe un aspecto concreto sobre el que el reciente crecimiento de AGI10 no provoca un incremento, sino todo lo contrario: los plazos de entrega. La consolidación de clientes importantes, como puede ser el grupo ADEO y en especial la multinacional Leroy Merlin, conlleva la imposición de una serie de estándares muy restrictivos, incluyendo unos plazos de entrega óptimos, los cuales deben ser cumplimentados sin afectar bajo ningún concepto a los plazos acordados con el resto de clientes. Uno de los mayores retos a los que nos debemos enfrentar para alcanzar dicho objetivo es, nuevamente, la resolución de incidencias. Cuando aparecen este tipo de situaciones es fundamental optimizar el tiempo de resolución, de forma que podamos evitar el incumplimiento de la planificación estimada y la ventana se encuentre lista para su expedición en el momento programado para ello. Mediante la implantación de nuestro sistema se proporcionan al usuario útiles para agilizar y optimizar la resolución de incidencias.
4. El último pilar sobre el que nos hemos apoyado para argumentar la realización de este proyecto es el ofrecimiento de una visualización interdepartamental. La aparición de un contratiempo en el proceso de producción de una ventana no afecta únicamente al propio departamento de producción, sino que toma un camino transversal. Con un sistema capaz de proporcionar dicha visualización, se permitiría a los distintos departamentos tomar decisiones dentro de su correspondiente ámbito, por ejemplo, abasteciendo un determinado material desde el departamento de compras, notificando un retraso a un cliente y elaborando una oferta compensatoria por parte del departamento comercial o de atención al cliente, o gestionando un cambio de prioridades y planificación desde el departamento de logística.

1.3 Estructura del documento

Dejando atrás este primer capítulo de introducción, presentaremos el problema objeto del trabajo en el próximo capítulo (2). Comenzaremos describiendo el estado del sistema pre-desarrollo (2.2), deficiencias de este (2.3), y objetivos propuestos (2.4), además de proporcionar al lector una pequeña introducción a la terminología de negocio utilizada (2.1). A continuación, abordaremos la metodología seleccionada para construir el proyecto (3), ahondando en el desarrollo de *software* ágil (3.1) y concretamente en su variante “programación extrema” (3.2). Posteriormente, y de forma muy breve expondremos las principales tecnologías y herramientas empleadas en el desarrollo de este proyecto (4). Una vez sentadas las bases del trabajo, desgranaremos el proceso *software* llevado a cabo (5), en el que detallaremos los roles de equipo definidos (5.1), los requisitos del sistema a implantar (5.2) y la planificación acordada (5.2), para posteriormente detallar la construcción del sistema (5.4) en base a las diversas etapas de implementación y pruebas (5.4.1 - 5.4.4). Finalmente, expondremos los trabajos futuros a realizar con el objetivo de completar el desarrollo (6), así como las conclusiones sacadas del mismo (7), donde en ambos casos se proporcionarán distintos enfoques o puntos de vista (6.1, 6.2 y 7.1, 7.2). Como punto final, se referenciará la bibliografía empleada a lo largo de este trabajo de fin de grado (8).

2 Presentación del problema

Dedicaremos este capítulo (2) a presentar el problema objeto de solución en este trabajo de fin de grado. En primer lugar, ofreceremos la idiosincrasia del problema (2.1), orientada a la terminología y conceptos particulares del mismo, de forma que se conforme una especie de diccionario sobre el flujo de negocio de AGI10 significativo para este proyecto. A continuación, expondremos la situación pre-desarrollo de la que partimos, los inconvenientes derivados de la misma, y las soluciones a implantar con el objetivo de resolverlos en las secciones 2.2, 2.3 y 2.4 correspondientemente.

2.1 Idiosincrasia

El objetivo de este apartado es presentar al lector la terminología propia del negocio que conste de relevancia para este proyecto, de forma que pueda retroceder a esta sección cuando se presente algún tipo de duda relacionada con la lectura del informe. A continuación, se presentan una serie de conceptos básicos orientados a la composición y diferentes formas que puede tomar la ventana a lo largo de su ciclo de vida en fábrica:

- La confirmación de una determinada versión de un presupuesto de cliente a pedido supone el punto inicial del que partiremos, ya que es en este instante cuando las ventanas contenidas en el mismo comienzan su ciclo de vida. Cada pedido se identifica mediante una asignación numérica única.
- A su vez, el contenido del pedido se distribuye en modelos. Cada modelo contenido en el pedido se identifica igualmente por un número de modelo, que va desde "1" hasta el número de modelos existentes en el pedido. Un *modelo* es la representación de una ventana bajo unas características únicas dentro del pedido, de forma que no pueden convivir dos modelos iguales dentro del mismo.
- Un modelo puede agrupar una o varias instancias. Siguiendo la misma lógica, cada instancia se identifica numéricamente dentro de cada modelo, desde "1" hasta el número de instancias abarcadas por el mismo modelo. Cada instancia representa una ventana física. Es por ello, que el identificador mediante el que podemos obtener unívocamente una determinada ventana es el conjunto conformado por [pedido, modelo, instancia].

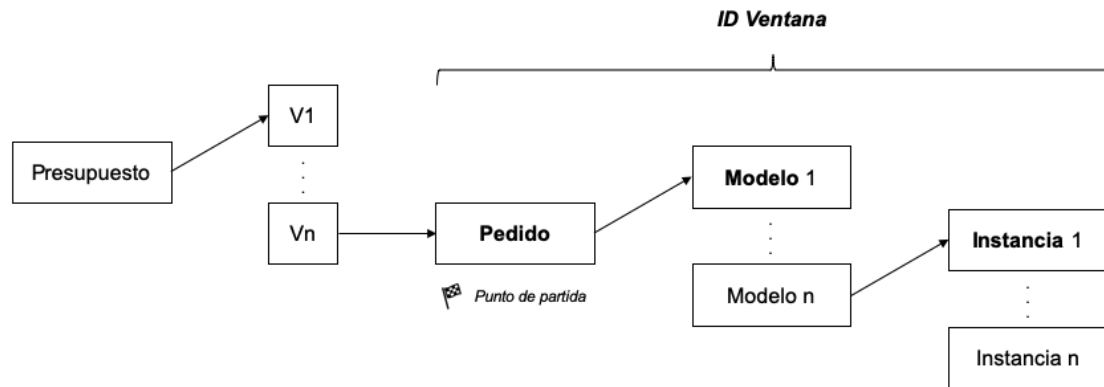


Ilustración 2.1: De presupuesto a instancia

- Para su fabricación, las instancias se distribuyen a lo largo de diversos lotes de producción. Un mismo lote de producción puede contener distintos ciclos, donde cada lote se identifica mediante una numeración única, y cada ciclo contenido en el mismo se numera desde “1” hasta el número de ciclos presentes en el lote. De esta forma, cada instancia puede ser asociada a una única combinación de [lote, ciclo] de producción. Distintas instancias de un mismo modelo (y, por tanto, pedido) pueden distribuirse en distintos lotes-ciclos de producción.
- Cuando una instancia pasa a formar parte de un lote-ciclo de producción, se generan todas las piezas que componen la misma. Cada pieza se identifica mediante un código de barras único. A su vez, una pieza puede formar parte o no de un cuadro de la instancia. Denominamos *cuadro* a una agrupación de piezas, el cual puede corresponderse con el marco o con alguna de las hojas de la instancia.

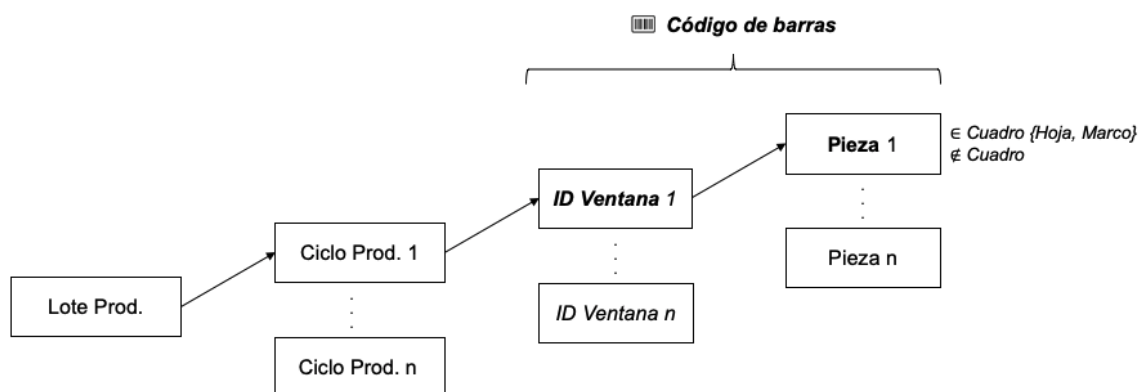


Ilustración 2.2: Composición lote-ciclo de producción

- Para su expedición, las instancias se distribuyen a lo largo de diversos lotes de envío, conformando una agrupación no necesariamente idéntica a la de los lotes de producción (de hecho, es el caso habitual). Los lotes de envío se identifican mediante una numeración única, donde cada instancia puede asociarse a uno de estos. Distintas instancias de un mismo modelo (y, por tanto, pedido) pueden distribuirse en distintos lotes de envío, aunque por norma general el envío de pedidos no se produce de forma incompleta.

2.2 Punto de partida

A continuación, presentamos la utilidad web mediante la cual un usuario registraba anteriormente una incidencia en el sistema:

LOTE 3088 (01) - Nueva Incidencia							
TIPOLOGÍA INCIDENCIA	Nº PEDIDO	NOM. VENTANA	Nº UNIDADES	DESCRIPCIÓN	ACCIÓN REPARADORA	SECCIÓN RESPONSABLE	TIEMPO MANO DE OBRA (en minutos)
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="button" value="INTRODUCIR INCIDENCIA"/>							

Ilustración 2.3: Formulario para el registro de incidencias pre-desarrollo

1. Para introducir una nueva incidencia en el sistema, el usuario seleccionaba el lote-ciclo de producción sobre el que quería reflejar la misma.
2. Una vez seleccionado, se habilitaba un sencillo formulario de incidencia, compuesto por una tipología basada en un selector acotado a una serie de valores y un conjunto de campos numéricos y de texto libre.
3. Cuando el usuario registraba la incidencia, la información asociada quedaba registrada en una tabla Access.
4. Para la lectura de la información se proporcionaba un archivo *mdb* compartido en red, para el que los distintos departamentos contaban con un acceso disponible.

2.3 Necesidad de cambio

A primera vista se observa que se trata de una introducción simple, poco profunda, y basada en campos de carácter libre sin ningún tipo de control aplicado, que facilitan los errores humanos e imposibilitan cualquier tipo de análisis sistemático, ya que no producen información cohesionada.

El único origen de introducción contemplado era el de lote-ciclo de producción, y su máximo nivel posible de detalle dentro del mismo era la especificación de un pedido (siempre y cuando el usuario tuviese en cuenta detallarlo correctamente).

A nivel de acceso, el archivo de base de datos Access compartido suponía un proceso arcaico, incómodo y que resultaba en problemas funcionales para el usuario, el cual sufría repetidos bloqueos y problemas de lectura.

2.4 Objetivos

El objetivo del desarrollo objeto de este informe es precisamente solventar todas las deficiencias a las que hemos hecho referencia:

1. Se persigue la creación de un sistema robusto en sus tripas o estructura, que sea capaz de generar información cohesionada, confiable y que impida la inconsistencia de datos.
2. Deberá tratarse de un sistema profundo, que permita llegar a un nivel de detalle e identificación de elementos considerable, al igual que flexible y adaptativo, siendo capaz de soportar la aparición de una situación de incidencia desde cualquier ámbito posible.
3. Sin embargo, la robustez y profundidad mencionadas no deben suponer ningún impedimento para la manejabilidad de la herramienta. Debemos tener en cuenta que un gran uso de la misma estará a cargo de los trabajadores situados en la cadena de montaje, que no tienen por que tener un amplio conocimiento informático, además de que su principal objetivo es la fabricación de ventanas y no el reporte de incidencias, por lo que debemos asegurarnos de proporcionar una interfaz de usuario muy intuitiva que proponga una interacción fluida, sin sacrificar ninguno de los anteriores requerimientos.
4. Por último, además de focalizarnos en una estructura y procesos de entrada de datos óptimos, debemos proporcionar un método eficiente de lectura y acceso a datos, resultando todo ello en un sistema cómodo, eficiente, versátil, y completo.

3 Metodología

La metodología de desarrollo *software* podría describirse como el marco de trabajo empleado para estructurar, organizar y controlar el proceso de desarrollo de un determinado sistema. En este capítulo justifiaremos la selección de una metodología ágil de desarrollo *software* para la realización de este proyecto (3.1), y concretamente la variante denominada “programación extrema” (3.2).

3.1 Desarrollo ágil

En primer lugar, debemos analizar cuál sería el enfoque apropiado para nuestro proyecto. Para ello hemos considerado tres pilares fundamentales:

1. El principal objetivo de nuestra organización es la producción de ventanas. Cualquier oportunidad que se presente por medio de un impacto favorable sobre la productividad es bienvenida, primando la brevedad en el tiempo de disposición de la herramienta sobre su propia completitud. Independientemente de contar con la forma final del producto, pero existiendo la posibilidad de que parte de su funcionalidad aporte un valor hasta el momento inexistente, se prioriza su pronta implantación con el fin de aprovecharnos de dichas ventajas, siguiendo progresiva y paralelamente con el resto de su desarrollo.
2. Otro punto a tener en cuenta es que se trata de un desarrollo de *software* intra-empresarial, dónde el cliente solicitante del proyecto y el equipo encargado de su desarrollo son diferentes departamentos de la misma empresa. En base a ello, consideramos que la naturalidad y la flexibilidad deben ser dos cualidades representativas del desarrollo. Partimos de una situación en la que se asegura que la involucración del cliente en el proyecto es total, ya que compartimos sus mismos objetivos, además de permitirnos una comunicación constante y fluida. La metodología seleccionada debe potenciar las ventajas que proporciona la gestión de un proyecto por la propia organización.
3. Por último, para que el proyecto desemboque en un resultado satisfactorio es importante mantener una actitud de tolerancia ante el cambio. Basándonos en la propia experiencia adquirida en otros desarrollos, hemos constatado como el cliente acaba modificando los requerimientos definidos en un primer momento una vez que entra en contacto con el producto, ya sea porque ha encontrado algún tipo de inconveniente o planteamiento incorrecto, o porque ha visto la posibilidad de añadir una nueva funcionalidad no contemplada previamente.

En base a los requerimientos expuestos, hemos corroborado que las metodologías ágiles de desarrollo *software* son la opción óptima para nuestro proyecto. El enfoque ágil, tal como se describe en [1], se basa en métodos de desarrollo incremental, centrado en el diseño rápido y presentando liberaciones frecuentes de *software*, donde el cliente esta altamente involucrado en el proceso de desarrollo. Estas metodologías, primeramente planteadas en la década de 1990, se sostienen sobre un conjunto de valores

acordados por sus principales valedores, y detallados sobre el denominado “manifiesto ágil” (véase [2]). Este manifiesto, redactado el 12 de febrero de 2001, presenta las siguientes valoraciones:

- **Individuos e interacciones** sobre procesos y herramientas.
- **Software funcionando** sobre documentación extensiva.
- **Colaboración con el cliente** sobre negociación contractual.
- **Respuesta ante el cambio** sobre seguir un plan.

Una vez presentado el enfoque de desarrollo de *software* ágil, analicemos algunas de sus características y constatemos cómo se adaptan plenamente a nuestra situación y requerimientos asociados:

1. Las metodologías ágiles proponen una entrega de *software* incremental, donde el producto no se desarrolla como una única unidad (modelo en cascada), sino que se presenta progresivamente en forma de nuevas funcionalidades incluidas en diferentes versiones del proyecto. Este principio se encuentra en concordancia con la orientación a la plena productividad de la empresa, donde se prioriza la pronta implantación de la herramienta a pesar de que todavía no cuente con el total de sus funcionalidades.

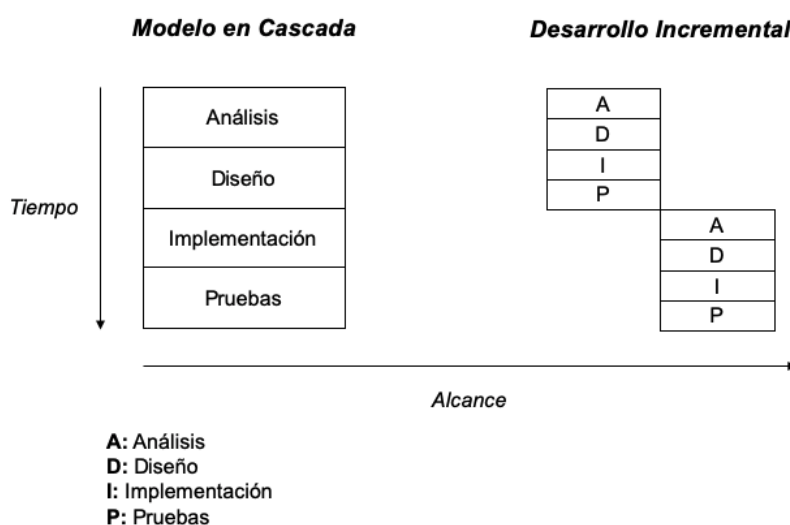


Ilustración 3.1: Comparativa modelo en cascada - desarrollo incremental

2. Otra de las bases de las metodologías ágiles es la participación del cliente. Como hemos expuesto anteriormente, al tratarse de un desarrollo intra-empresarial, la involucración del cliente es una de las ventajas con las que contamos y está garantizada desde el primer momento, por lo que el soporte de este principio por parte de los métodos ágiles es fundamental. Además, los desarrollos de sistemas a medida en una empresa, donde no existen muchas reglas ni regulaciones externas a las que se vea sometido el *software*, constituyen un gran foco de acción para este tipo de metodologías.

3. El desarrollo de *software* ágil implica la adaptación al cambio. Se espera que a lo largo del ciclo de vida del proyecto los requerimientos del sistema se vean alterados, por lo que se busca el diseño de un sistema propenso a la adaptación. Previamente hemos reflejado como esta es una práctica habitual en las herramientas desarrolladas por y para nuestra empresa, por lo que un diseño orientado a soportar el cambio unido a un conjunto de entregas incrementales que sean capaces de sacar a la luz dichas variaciones en los requerimientos, tal como se propone en las metodologías de desarrollo ágil, conforman medidas óptimas para garantizar el éxito del proyecto en nuestra organización.
4. Otra de las propuestas de estas metodologías es la de sobreponer la persona al proceso, permitiendo al equipo desarrollar sus propias formas de trabajo sin caer en la imposición de las mismas. Esta característica toma una gran relevancia para con nuestro proyecto, donde a pesar de que se requiere un trabajo transversal con el resto de departamentos para la definición del sistema, el desarrollo *software* al completo va a ser cargo de una única persona. De esta forma, no tiene ningún valor una burocratización del proceso, donde solo obtendríamos una pérdida de fluidez en el trabajo que repercutiría en un retraso en el tiempo de implantación de la herramienta.

3.2 Programación Extrema (XP)

Llegados a este punto es momento de detallar las diferentes prácticas para el desarrollo ágil de *software* seleccionadas, con el objetivo de estructurar, organizar y controlar el proceso de desarrollo de nuestro proyecto. Hay que tener en cuenta que los métodos ágiles están normalmente definidos para un equipo de trabajo, reducido, pero un equipo, al fin y al cabo. En nuestro caso se trata de un desarrollo de *software* individual, por lo que no se ha encontrado una metodología concreta que se adapte totalmente a nuestro propósito. Dicho esto, muchas de las prácticas planteadas en la programación extrema, o XP, son altamente aplicables a nuestro marco de trabajo, por lo que se ha considerado una solución que implemente una buena parte de las mismas, incluyendo un conjunto de alteraciones necesarias para adaptar XP a nuestro proyecto.

La programación extrema, formulada por el ingeniero de *software* y uno de los firmantes del manifiesto ágil, Kent Beck [3], es probablemente el método mas destacado de desarrollo ágil, denominado de dicha forma debido al planteamiento de llevar las prácticas reconocidas a niveles extremos. A continuación, presentamos dicha propuesta de prácticas que conforman XP, así como su adaptación a nuestro propio escenario:

- **Planeación incremental.**

- Los requerimientos se especifican en tarjetas de historia, o *story cards*. Desarrolladas de forma conjunta con el cliente, contienen las necesidades de este. Una vez diseñadas, el equipo de desarrollo descompone las mismas en un conjunto de tareas, que constituyen la unidad principal de implementación.
- Las tarjetas de historia y sus correspondientes subdivisiones en tareas implementables aportan un plus de informalidad a la relación con el cliente, además de facilitar la comprensión y comunicación entre ambas partes sobre el clásico documento de requisitos. Creemos que en el entorno de trabajo que nos compete, es una gran medida para agilizar el desarrollo del proyecto, por lo que haremos uso de ella.

- **Liberaciones pequeñas.**

- Partiendo de una primera liberación con funcionalidad reducida pero útil, se van sucediendo de forma frecuente nuevas liberaciones que vayan incorporando funcionalidad a la primera.
- Como hemos expuesto anteriormente en diferentes puntos, AGI10 prioriza la pronta funcionalidad que pueda agregar valor de negocio tan pronto como esta este disponible. Las liberaciones frecuentes de *software* sustentan este planteamiento y será una de las prácticas que apliquemos.

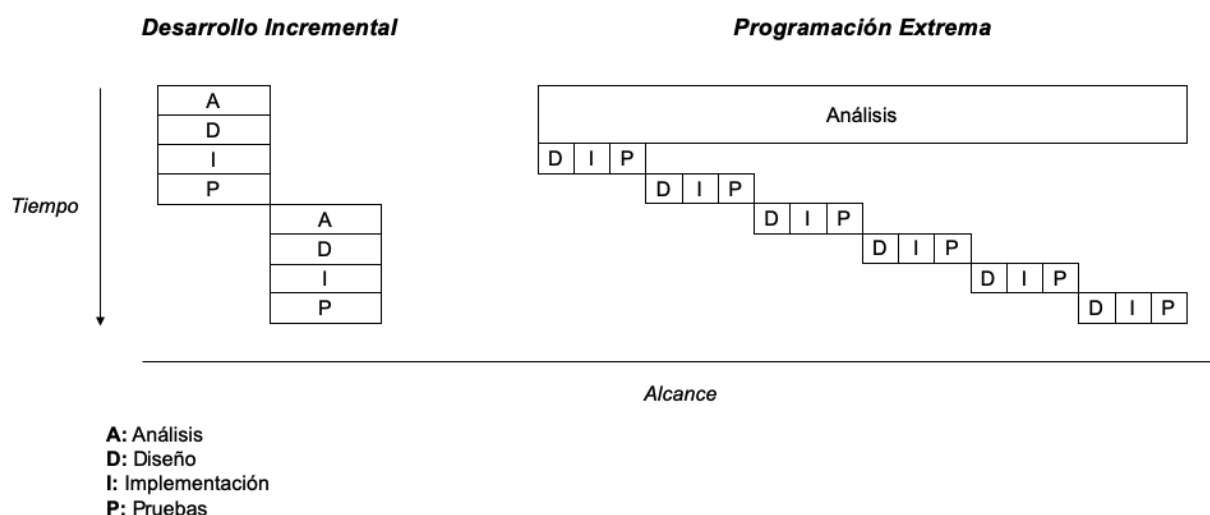


Ilustración 3.2: Comparativa desarrollo incremental - Programación Extrema

- **Diseño simple.**

- Realización de un diseño que, soportando toda la funcionalidad requerida, mantenga la mayor simplicidad posible.
- Mediante esta práctica se conseguirá una reducción en los tiempos de desarrollo y, por ende, de despliegue de la liberación. Facilitará que el producto pueda aportar valor real desde etapas tempranas del proyecto, por lo que nos será de gran utilidad.

- **Desarrollo de la primera prueba.**

- XP propone que, en lugar de escribir un código y a posteriori realizar pruebas sobre el mismo, las pruebas se elaboren antes de la propia implementación. Permite la ejecución de pruebas conforme se desarrolla el código y descubrir problemas en tiempo real.
- Es una de las innovaciones mas interesantes de XP. Conlleva que el equipo encargado de la implementación comprenda ampliamente la especificación, de modo que tanto las posibles ambigüedades y omisiones de información en las tarjetas de historia como las dudas por parte del desarrollador respecto al flujo presentado en las mismas queden resueltas desde el primer instante. Práctica muy ventajosa para nuestra organización y proyecto, donde el departamento

de sistemas no tiene por qué estar, en primera instancia, totalmente familiarizado con los conceptos relativos a la fabricación de la ventana.

- **Refactorización.**

- Refactorización continua del código de forma que este se vea optimizado constantemente.
- Basándonos en experiencias anteriores, la necesidad inminente de una nueva funcionalidad para el sistema no permite la involucración del desarrollador en la optimización de un código ajeno que ya funciona correctamente. En base a ello, hemos valorado para este proyecto un equilibrio entre la refactorización ocasional y la producción de código óptimo en primera instancia.

- **Cliente en sitio.**

- La programación extrema considera de vital importancia que un representante del cliente se encuentre involucrado a tiempo completo en el proyecto, formando parte del equipo XP. Este sería el responsable de trasladar los requerimientos del sistema de una parte (cliente) a la otra (equipo).
- Es una práctica que somos capaces de cumplimentar, facilitada por el carácter intra-empresarial del desarrollo. Contaremos con una persona del departamento de producción con un perfil semi-tecnológico que desempeñara el rol propuesto por XP.

- **Ritmo sustentable.**

- No se recomienda invertir grandes cantidades de tiempo extra en el proceso, ya que frecuentemente estas desembocan en una pérdida de productividad y calidad en el código.
- Es una práctica que naturalmente se adapta a nuestra forma de trabajo en AGI10. Hay que tener en cuenta que no somos un departamento orientado estrictamente a la producción de *software*, sino que nuestra área de trabajo es mucho mayor. De esta forma, tendremos espacios temporales concretos a lo largo del día destinados al proyecto, lo que va en contraposición con la inversión de grandes cantidades de tiempo extra.

- **Integración continua.**

- Integración inmediata de las tareas en el sistema tal como estas se vayan completando.
- Esta práctica se encuentra mas orientada a aquellos proyectos que involucran un grupo de desarrolladores, donde subconjuntos de estos trabajan sobre distintas áreas del código, de forma que se pueda verificar rápidamente que el trabajo de un miembro del equipo no ha afectado negativamente sobre el resto de funcionalidades. A pesar de que en nuestro caso se trate de un desarrollo *software* individual, consideramos oportuna su puesta en práctica.

- **Programación en pares y propiedad colectiva.**

- XP propone el trabajo en pares para proporcionar apoyo y comprobaciones cruzadas, así como la implicación del par a lo largo de todo el código, impidiendo la aparición de islas de experiencia.
- Ambas prácticas constituyen un enfoque para el desarrollo grupal del proyecto. En nuestro caso esto no se aplicable, ya que el proceso *software* está encargado a una única persona. De esta forma, ninguna de estas prácticas incorporarse al proyecto.

4 Tecnologías y herramientas

4.1 HTML5 + CSS3 + JavaScript + AJAX

HyperText Markup Language, o HTML [4], es un lenguaje de marcado para el desarrollo de páginas web. Hemos empleado la versión 5, para la cual la mayoría de navegadores ofrecen soporte.

Cascading Style Sheet, o CSS [4], es el lenguaje encargado de dotar de estilo a los documentos HTML. Se ha utilizado la versión 3, última evolución de este lenguaje.

JavaScript [5], es un lenguaje de programación interpretado, utilizado principalmente en el lado del cliente y capaz de interactuar con la página a través del Document Object Model (DOM).

Finalmente, Asynchronous JavaScript And XML o Ajax [4], representa una conjunción de JavaScript y XML que posibilita el intercambio asíncrono de datos entre el cliente y servidor.

4.2 PHP + Swift Mailer

Hypertext Preprocessor, o PHP [6], es un lenguaje de programación diseñado para el lado del servidor y cuyo principal objetivo es la generación dinámica de contenido web. Para el desarrollo del proyecto hemos trabajado con PHP7, última versión del lenguaje.

Swift Mailer [7], por su parte, es una librería basada en componentes que posibilita el envío de correos electrónicos a través de PHP.

4.3 Microsoft SQL Server + Transact-SQL

Microsoft SQL Server [8] es un sistema de gestión de base de datos (BD) relacional, donde Transact-SQL [9], o T-SQL, es el lenguaje de desarrollo que conforma el medio de interacción con el servidor mediante el envío de sentencias y declaraciones.

4.4 Microsoft Office + Microsoft PowerBI

Microsoft Office [10] es una suite ofimática que permite la creación de todo tipo de documentos, empleada para la construcción de este informe a través de su versión de 2019.

Igualmente, hemos trabajado con Microsoft Power BI [11], el cual conforma un servicio de análisis empresarial para la visualización interactiva de datos, con el que hemos obtenido indicadores e informes.

5 Proceso software

En este capítulo (5) detallaremos el ciclo de vida del desarrollo *software* realizado, basado en los principios de la metodología seleccionada: la programación extrema. Para ello, identificaremos los distintos roles del equipo de trabajo (5.1), así como las historias de usuario (5.2) que definirán los requisitos del sistema a implantar. A continuación, desgranaremos la planificación del proyecto (5.3) distribuido en diferenciadas liberaciones de *software*, para finalmente, presentar el trabajo realizado en cada una de las mismas, enfocado principalmente sobre las tareas de implementación y pruebas (5.4).

5.1 Roles

Para que un desarrollo de *software* culmine de forma satisfactoria, es fundamental crear una estructura de trabajo donde la comunicación entre los distintos integrantes del equipo sea fluida y eficiente. En XP, los roles de equipo proponen un trabajo en grupo para asegurar el éxito en el trabajo individual de cada componente.

Tal como sea plantea en [12], en XP, la definición de estos roles no responde a un conjunto de reglas rígidas. El objetivo de estos es potenciar la contribución de cada integrante, de forma que repercuta sobre el éxito del grupo. Con este objetivo en mente y centrándonos en el proyecto que nos compete, se formuló un equipo con su correspondiente distribución de roles adaptados a las necesidades particulares del desarrollo.

Los integrantes del equipo XP y los roles desempeñados a cargo de cada uno de los mismos se listan a continuación:

- **Responsable del departamento de sistemas:** El responsable del departamento de sistemas tomaría un rol de gestor. Su principal labor es la de coordinación entre el cliente y el programador. En nuestro caso particular, donde el programador no tiene asegurada una dedicación completa de su jornada laboral al proyecto, debido a la prioridad e inmediatez de las tareas orientadas a la producción en fábrica, la correcta gestión de los tiempos de trabajo toma una relevancia especial. Este coordinador debe asegurar al programador los periodos de dedicación al proyecto, además de planificar las reuniones entre este y el cliente. En nuestro caso, también puede proporcionar soluciones técnicas al programador en situaciones comprometidas.
- **Integrante del departamento de sistemas:** Se corresponde con el escritor del presente informe, y estaría a cargo de dos roles distintos: programador y tester. El *programador* podría definirse como el elemento central o esencia del equipo, encargado de valorar las historias de usuario, descomponerlas en tareas implementables, además de desarrollar todo el código que soporte las distintas funcionalidades y la definición de las pruebas unitarias. Por otro lado, tendríamos el rol *tester*; mientras que el programador se encarga del diseño y comprobación de las pruebas unitarias, el enfoque del tester es algo mas general, estando a cargo de las pruebas funcionales y garantizado por

medio de las mismas la estabilidad y aceptación del desarrollo. En nuestro proyecto ambos roles son desempeñados por el mismo integrante del equipo, por lo que podría considerarse como un trabajo continuo.

- **Responsable del departamento de operaciones:** El rol de cliente será representado por el responsable del departamento de operaciones. Podría decirse que la persona seleccionada tiene un perfil un tercio tecnológico, y dos tercios orientados al proceso de fabricación de la ventana, lo que supone un nexo perfecto entre los usuarios finales del sistema y el desarrollador. Además, cuenta con experiencia en sistemas de similares características, por lo que tiene un concepto muy claro de las necesidades funcionales de la herramienta a implantar, así como los conocimientos técnicos suficientes para establecer una comunicación fluida con el programador.
- **Jefe de producción:** Por último, el equipo contará con el apoyo del jefe de producción, a modo de consultor. La tarea del *consultor* es proporcionar ayuda al equipo, con el objetivo de resolver problemas específicos. En nuestro caso, esta ayuda se emplearía para resolver dudas puntuales relacionadas con el ciclo de vida de la ventana en fábrica que el cliente no sea capaz de determinar a ciencia cierta.

5.2 Historias de usuario

En la programación extrema, las historias de usuario constituyen la alternativa a los tradicionales documentos de requisitos. Una de las principales diferencias entre ambas técnicas puede observarse en el nivel de detalle proporcionado, donde, de acuerdo a [13], las historias de usuario deben ser únicamente lo suficientemente explícitas como para ser capaz de estimar, con un riesgo menor, el tiempo que esta tardará en implementarse. Escritas en un lenguaje natural, y prescindiendo de detalles referentes a las distintas tecnologías involucradas, estas se centran plenamente en las necesidades del usuario.

En esta sección detallaremos todas las historias de usuario abarcadas por nuestro proyecto. Cada historia cuenta con un identificador, una descripción y un conjunto de criterios de aceptación. Se recomienda que la descripción de la misma sea breve, y basada en una estructura “como usuario, quiero algo, por el siguiente motivo”. Mientras elaborábamos estas historias de usuario, observamos cómo esta regla concreta imponía una restricción innecesaria en la expresión de las mismas, por lo que a petición del cliente se tomo una especificación mas fluida, que facilitara el entendimiento entre este y el programador. Por otro lado, tenemos los criterios de aceptación, empleados para determinar cuándo una historia de usuario puede darse como finalizada, así como para clarificar el contexto de la misma.

A continuación, se listan el total de las historias de usuario implementadas en el desarrollo:

- **Historia 1: Orígenes de incidencia.**
 - **Descripción:** Una incidencia podrá especificarse desde los siguientes tipos de origen: planta, lote de producción, lote de envío y pedido. Donde los tres últimos (lote de producción, lote de envío y pedido) hacen referencia a organizaciones lógicas de ventanas, mientras que “planta” identifica un origen de carácter genérico, donde no se identifican ventanas de forma específica.
 - **Criterios de aceptación:**
 - 1) El usuario tendrá la capacidad de seleccionar el tipo origen de la incidencia entre los descritos anteriormente (planta, lote de producción, lote de envío y pedido).

- 2) Para aquellos tipos de orígenes que conformen una agrupación lógica de ventanas, el usuario deberá proporcionar su identificador numérico.
- 3) En base a dicho identificador, se deberá comprobar si el origen especificado por el usuario existe en el sistema.

- **Historia 2: Composición de incidencia.**

- **Descripción:** Todas las incidencias contarán con un conjunto de información común, donde parte de ella será rellenada a través del usuario mediante un formulario, y la restante se capturará automáticamente. La primera parte se conformará con los campos: tipología de incidencia, acción reparadora, puesto de detección y descripción de la incidencia, mientras que la segunda a partir de: identificador de usuario y fecha/hora de apertura de incidencia.
- **Criterios de aceptación:**
 - 1) Se presentará un formulario con los campos a rellenar mencionados (tipología de incidencia, acción reparadora, puesto de detección y descripción de la incidencia).
 - 2) Los campos “tipología de incidencia” y “acción reparadora” son campos de carácter obligatorio, mientras que “puesto de detección” y “descripción” son campos de carácter opcional.
 - 3) El campo “tipología de incidencia” dependerá del origen seleccionado, pudiendo distintos orígenes mostrar diferentes tipologías de incidencia.
 - 4) El campo “acción reparadora” dependerá de la tipología de incidencia seleccionada, pudiendo distintas tipologías de incidencia mostrar diferentes acciones reparadoras.
 - 5) Por otro lado, el sistema capturará implícitamente los campos “identificador de usuario” y “fecha/hora de apertura” de incidencia.

- **Historia 3: Especificación de máquina.**

- **Descripción:** Si el usuario selecciona una tipología de incidencia que implique una máquina de producción, se habilitará un nuevo campo “máquina afectada” para su especificación.
- **Criterios de aceptación:**
 - 1) Si el usuario selecciona una tipología que implique una máquina, se habilitará automáticamente un nuevo campo “máquina afectada”.
 - 2) El campo “puesto” pasará a ser de carácter obligatorio.
 - 3) El campo “máquina afectada” dependerá del puesto seleccionado, pudiendo distintos puestos mostrar diferentes máquinas de producción.
 - 4) El campo “máquina afectada” será un campo de carácter obligatorio.

- **Historia 4: Asociación de incidencia a contenido.**

- **Descripción:** Para aquellas incidencias cuyo origen se corresponda con una organización lógica de ventanas (lote de producción, lote de envío o pedido), el usuario deberá especificar qué parte del contenido de dicho origen se ha visto afectado por la incidencia en cuestión.
- **Criterios de aceptación:**
 - 1) Las vías de asociación de las que dispondrá el usuario para establecer el contenido afectado variarán en función de la acción reparadora seleccionada, pudiendo distintas acciones reparadoras mostrar distintas opciones de asociación.
 - 2) En el mejor de los casos, se permitirá una asociación de la incidencia al origen completo, o a contenido concreto del mismo. En caso de seleccionar una asociación a contenido concreto, se permitirá una selección múltiple de elementos de las naturalezas "pedido", "modelo" e "instancia" en base al contenido del propio origen, pudiendo una única incidencia afectar a elementos de distintas naturalezas. Como hemos mencionado, la restricción de estos caminos de asociación variará en función de la acción reparadora seleccionada.
 - 3) En caso de una asociación a un elemento pedido o modelo con una única ventana contenida, el sistema deberá realizar una conversión automática a un elemento instancia.

- **Historia 5: Notificación de apertura.**

- **Descripción:** En función de la tipología de incidencia seleccionada, se notificará el registro de la misma por medio de un correo electrónico.
- **Criterios de aceptación:**
 - 1) Si el usuario da de alta una incidencia cuya tipología conlleve notificación, el sistema enviará automáticamente un correo electrónico a los responsables adecuados.
 - 2) El contenido del correo electrónico deberá reflejar la información completa de la incidencia en cuestión.

- **Historia 6: Listado de incidencias registradas.**

- **Descripción:** Se proporcionará un listado que refleje la información correspondiente a las distintas incidencias registradas en el sistema.
- **Criterios de aceptación:**
 - 1) Se presentará un listado dónde el usuario tendrá la posibilidad de consultar las incidencias registradas. Cada incidencia se representará como una nueva entrada en el listado. Estas entradas mantendrán toda la información relativa a la incidencia: identificador, usuario y fecha de apertura, origen, tipología, acción reparadora, puesto, máquina si se aplica, descripción y tipo de asociación, así como el contenido afectado por la misma.

- **Historia 7: Paginación del listado de incidencias.**
 - **Descripción:** Para facilitar su visualización, se distribuirán las entradas del listado de incidencias en distintas páginas de tamaño variable.
 - **Criterios de aceptación:**
 - 1) El usuario dispondrá de un control mediante el que podrá especificar cuántos registros desea acumular en cada visualización.
 - 2) Se habilitará una paginación que distribuya el total de entradas resultantes en visualizaciones agrupadas en base al número de registros estipulados por el usuario.

- **Historia 8: Ordenación del listado de incidencias.**
 - **Descripción:** Para facilitar su visualización, se habilitará la ordenación de las entradas del listado de incidencias en base a diferentes atributos clave.
 - **Criterios de aceptación:**
 - 1) El usuario tendrá la posibilidad de realizar una ordenación ascendente/descendente de las entradas por los siguientes atributos: apertura, tipología y acción reparadora.

- **Historia 9: Filtrado del listado de incidencias.**
 - **Descripción:** Será posible acometer un filtrado de las entradas del listado de incidencias en base a un conjunto de opciones basadas en las características de las propias incidencias.
 - **Criterios de aceptación:**
 - 1) Se presentará un panel donde el usuario tendrá la capacidad de realizar un filtrado de las entradas en base a las siguientes características de incidencia: identificador, fecha de apertura, tipología de la incidencia, acción reparadora, puesto de detección, máquina, descripción y contenido afectado.
 - 2) Las opciones de filtrado descritas podrán seleccionarse de forma simultánea.

- **Historia 10: Gestión de filtros.**
 - **Descripción:** Una vez confeccionado un filtro, se permitirá el almacenamiento del mismo de cara a su uso en futuros accesos al sistema.
 - **Criterios de aceptación:**
 - 1) Se habilitará una opción de guardado en el panel de filtrado, donde el usuario podrá asignarle un nombre al filtro confeccionado en base a la selección de las distintas opciones presentes.
 - 2) Una vez almacenado, se añadirá a la lista de filtros confeccionados por el usuario. En cualquier momento el usuario podrá seleccionar una de estas composiciones, respondiendo automáticamente el listado de incidencias a los parámetros definidos en el filtro seleccionado.

- 3) Para un filtro almacenado, el usuario podrá definir si este será cargado automáticamente en cada acceso a la utilidad.

- **Historia 11: Unificación de utilidades.**

- **Descripción:** Las utilidades de recolección y búsqueda de incidencias se unificarán en un único módulo destinado a componer un gestor de calidad centralizado.
- **Criterios de aceptación:**
 - 1) Se sustituirán los accesos individualizados y aislados por un único módulo centralizado que mantenga ambas utilidades.
 - 2) Este módulo centralizado deberá diseñarse de forma que pueda mantener futuras nuevas aplicaciones con el objetivo de completar la funcionalidad del gestor de calidad.

- **Historia 12: Gestión de incidencias.**

- **Descripción:** Se permitirá la gestión de un conjunto de incidencias de carácter delicado, pudiendo el responsable acometer tanto una serie de acciones especiales, como su cierre cuando se considere que la misma ha sido resuelta.
- **Criterios de aceptación:**
 - 1) Para las incidencias cuya tipología conlleve una recepción de faltante, se habilitará en el listado de incidencias una funcionalidad específica mediante la cual el responsable podrá definir una fecha estimada de recepción para el faltante correspondiente a la entrada deseada.
 - 2) Para las incidencias cuya acción reparadora conlleve una reparación, se habilitará en el listado de incidencias una funcionalidad específica mediante la cual el responsable podrá consultar el contenido concreto a reparar correspondiente a la entrada deseada. Se requiere que se añadan nuevas opciones de búsqueda en la asociación de incidencias a contenido concreto, respondiendo a las naturalezas: pieza de marco, pieza de hoja, pieza de travesaño, marco y hoja.
 - 3) Para las incidencias cuya tipología conlleve una actuación en máquina, se habilitará en el listado de incidencias una funcionalidad específica mediante la cual el responsable podrá definir un tiempo estimado y real de actuación para la máquina correspondiente a la entrada deseada.
 - 4) En el panel de filtrado, se añadirá una nueva opción por la cual podrá seleccionarse la visualización de incidencias que impliquen una recepción de faltante, reparaciones o actuación en máquina.
 - 5) Se proporcionará una funcionalidad en el listado de incidencias mediante la cual el responsable pueda marcar las incidencias de estas características como resueltas.

- **Historia 13: Jerarquía de permisos.**

- **Descripción:** El sistema tendrá la capacidad de permitir o restringir los distintos accesos y funcionalidades en función del usuario que acceda a la herramienta.
- **Criterios de aceptación:**
 - 1) Para que un usuario pueda acceder a una determinada funcionalidad se requerirá de un permiso explícito para el mismo. En caso contrario se restringirá por defecto.

- **Historia 14: Estado de incidencia.**

- **Descripción:** Se añadirá a la información de incidencia accesible en el listado una nueva propiedad por la que se notificará el estado actual de la incidencia en base a las nuevas posibilidades de gestión.
- **Criterios de aceptación:**
 - 1) Se habilitará en el listado de incidencias una propiedad “estado” para cada entrada del mismo. Las incidencias que conlleven una gestión (faltante, reparación o actuación en máquina), la propiedad “estado” podrá tomar los valores “en curso” o “resuelta”. Para las incidencias que no conlleven una gestión y su correspondiente resolución, el estado simplemente será de “notificado”.
 - 2) En el panel de filtrado, se añadirá una nueva opción por la cual podrá seleccionarse entre incidencias en curso y resueltas.

- **Historia 15: Notificación de resolución.**

- **Descripción:** En función de la tipología de incidencia seleccionada, se notificará la resolución de la misma por medio de un correo electrónico.
- **Criterios de aceptación:**
 - 1) Si el usuario cierra una incidencia cuya tipología conlleve notificación, el sistema enviará automáticamente un correo electrónico a los responsables adecuados.
 - 2) El contenido del correo electrónico deberá reflejar la información completa de la incidencia en cuestión.

5.3 Planificación

Una vez redactadas las distintas historias de usuario podemos poner en marcha la planificación del proyecto, para la cual la programación extrema, como se aprecia en [14], propone una estructura muy definida, esquematizada sobre el siguiente gráfico:

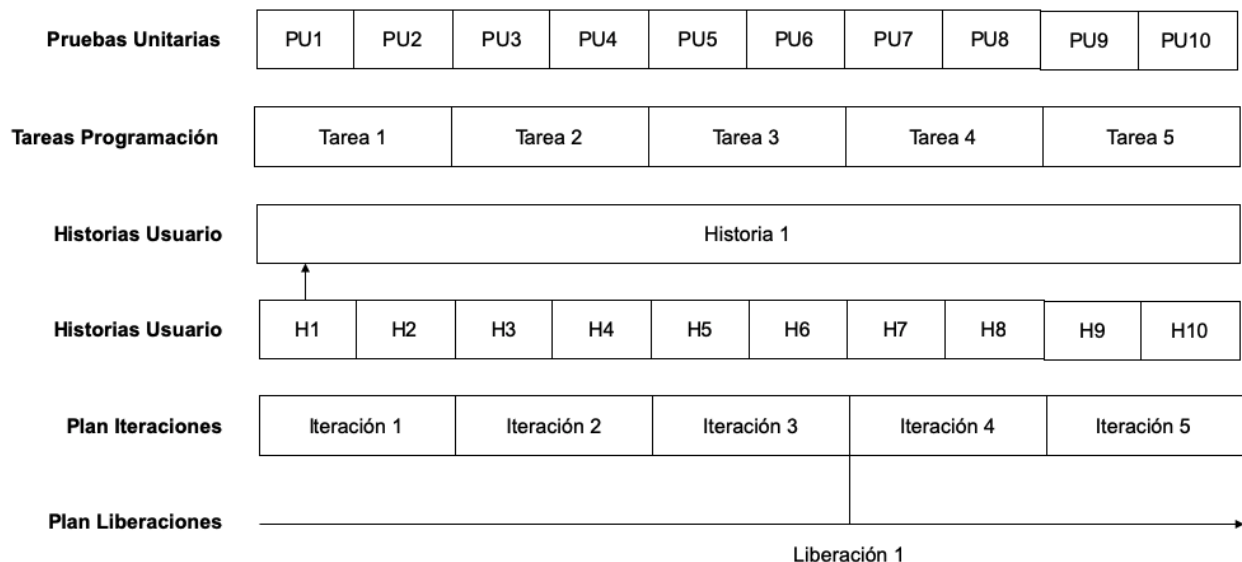


Ilustración 5.1: Jerarquía de planificación en XP

El nivel más alto de la jerarquía de planificación en XP es ocupado por las liberaciones. El plan de liberaciones especifica qué historias de usuario serán implementadas en cada liberación, además de proporcionar una fecha para la misma. Para ello, se celebra una reunión con todos los integrantes del equipo, donde los desarrolladores estiman tiempos de implementación para cada una de las historias, y los clientes asignan prioridades a estas.

A posteriori se presenta el plan de iteraciones, por el cual cada una de las liberaciones acordadas se divide en un conjunto de iteraciones. A diferencia del plan de liberaciones, donde la reunión se lleva a cabo al principio del proyecto, cada iteración se planifica en el comienzo de la misma. Al igual que las liberaciones, una iteración abarca un conjunto de historias de usuario a ser implementadas, las cuales son detalladas a fondo por el usuario en este mismo punto, siguiendo el orden de prioridad definido previamente en la reunión de planificación.

Una vez definidas las iteraciones para una determinada liberación, los desarrolladores descomponen cada historia de usuario en un conjunto de tareas implementables, que, a diferencia de las primeras, no están escritas completamente en un lenguaje de alto nivel, sino que permiten la conjugación con un enfoque más técnico. Estas tareas proporcionan las pruebas de unidad, mientras que las historias de usuario proveen las pruebas de aceptación.

Tanto al final de una liberación como al final de una iteración se proporciona funcionalidad al usuario, sin embargo, la primera hace referencia a un despliegue del sistema en el entorno de producción,

mientras que las iteraciones resultan en funcionalidad para el cliente, de forma que este pueda ir valorando el desarrollo del proyecto e ir proporcionando *feedback* al equipo de desarrollo en base a las historias implementadas.

A continuación, presentamos el plan de liberaciones e iteraciones que fue ejecutado en el desarrollo de este proyecto:

- **Liberación 1: Recolección de Incidencias (4 semanas - 25 marzo 2019)**

- Primera Iteración (1 semana - 4 marzo 2019)
 - **Historia 1:** Orígenes de incidencia.
 - **Historia 2:** Composición de incidencia.
- Segunda Iteración (2 semanas - 11 marzo 2019)
 - **Historia 4:** Asociación de incidencia a contenido
- Tercera Iteración (1 semana - 25 marzo 2019)
 - **Historia 3:** Especificación de máquina.
 - **Historia 5:** Notificación de apertura.

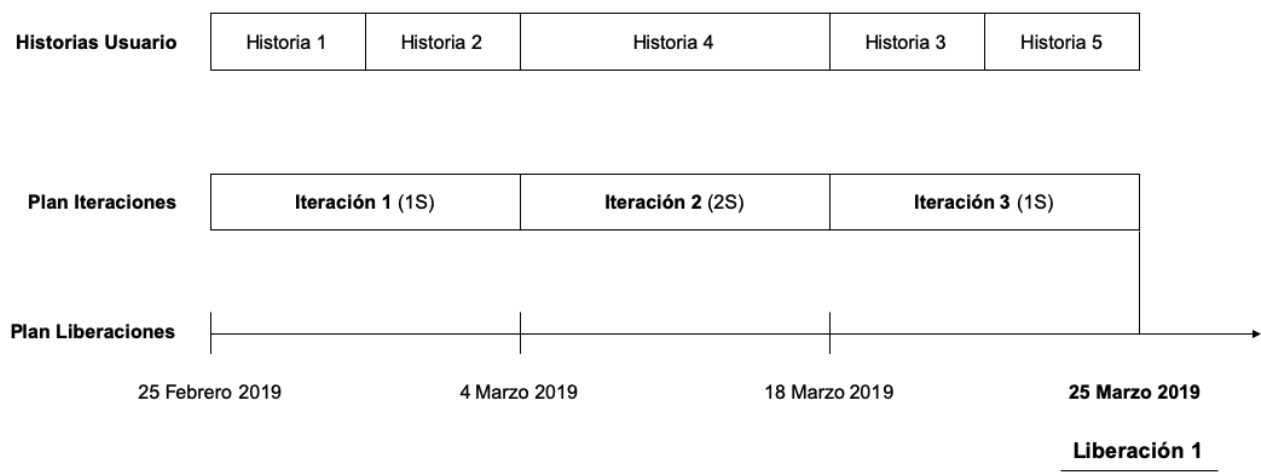


Ilustración 5.2: Planificación Liberación 1 - Recolección de incidencias

- **Liberación 2: Búsqueda de incidencias (4 Semanas - 22 abril 2019)**

- Cuarta Iteración (1 semana - 1 abril 2019)
 - **Historia 6:** Listado de incidencias registradas.
 - **Historia 7:** Paginación del listado de incidencias.
- Quinta Iteración (2 semanas - 15 abril 2019)
 - **Historia 9:** Filtrado del listado de incidencias.
- Sexta Iteración (1 semana - 22 abril 2019)
 - **Historia 8:** Ordenación del listado de incidencias.
 - **Historia 10:** Gestión de filtros.

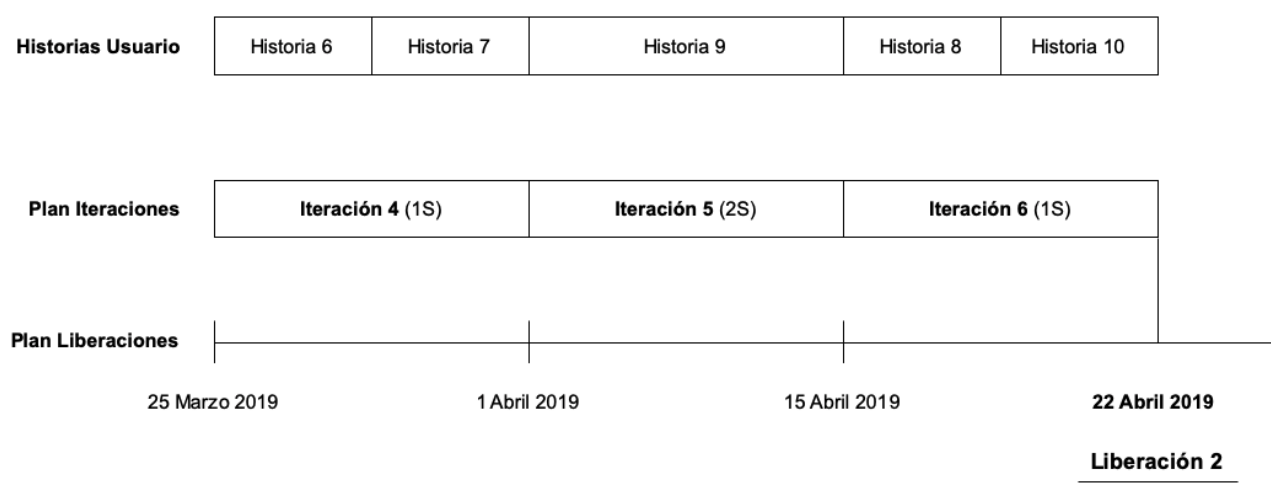


Ilustración 5.3: Planificación Liberación 2 - Búsqueda de incidencias

- **Liberación 3: Unificación + Gestión de incidencias (4 semanas - 20 mayo 2019)**

- Séptima Iteración (1 semana - 29 abril 2019)
 - **Historia 11:** Unificación de utilidades.
 - **Historia 13:** Jerarquía de permisos.
- Octava Iteración (2 semanas - 13 mayo 2019)
 - **Historia 12:** Gestión de incidencias.
- Novena Iteración (1 semana - 20 mayo 2019)
 - **Historia 14:** Estado de incidencia.
 - **Historia 15:** Notificación de resolución.

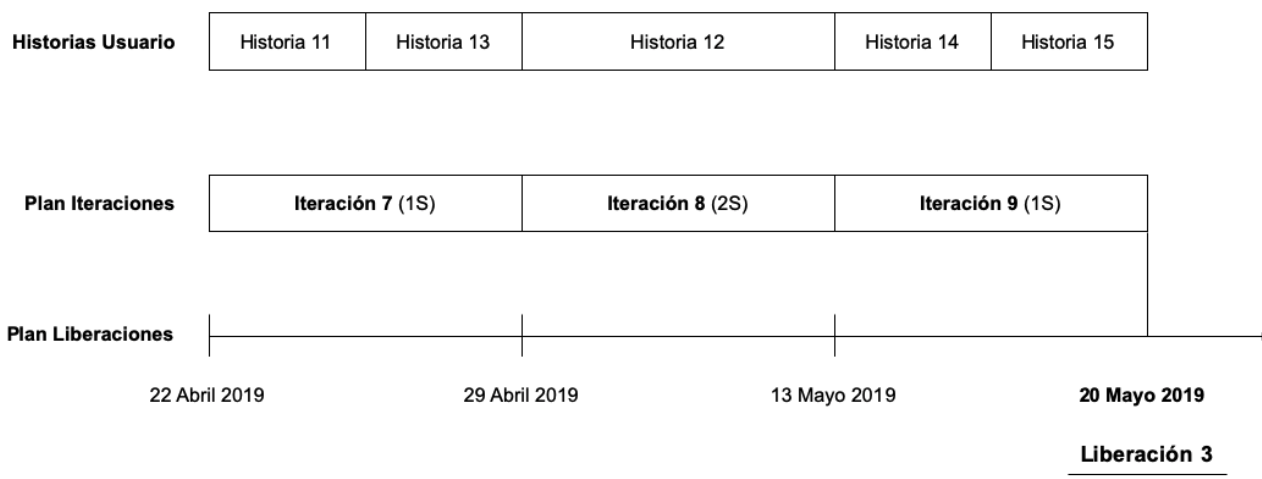


Ilustración 5.4: Planificación Liberación 3 - Unificación + Gestión de incidencias

5.4 Liberaciones

El resto del capítulo sobre el proceso *software* mostrará el trabajo realizado para desarrollar cada una de las liberaciones planificadas. Mientras que nuestra primera intención era diseccionar el capítulo conforme a cada una de ellas con el objetivo de proporcionar una visión progresiva del desarrollo donde se pudiese apreciar en cada liberación las decisiones tomadas para adaptar el sistema a las nuevas funcionalidades requeridas, debido a la longitud del informe nos hemos visto obligados a tomar un enfoque transversal, donde para cada sección ejemplificaremos el trabajo realizado en base a desarrollos de una u otra liberación.

Cada una de las secciones se estructurará siguiendo el mismo patrón, orientado principalmente a las fases iterativas y entrelazadas de implementación y pruebas, siendo consecuentes con los principios de la programación extrema. En primer lugar, mostraremos la descomposición en tareas implementables de las historias de usuario planificadas para una determinada iteración (5.4.1). A partir de estas, generaremos sus correspondientes pruebas unitarias (5.4.2), de forma que el desarrollo y *testeo* del sistema se apoyen mutua y constantemente. Una vez que contemos con este conjunto de tareas y pruebas unitarias, analizaremos la implementación (5.4.3) práctica de la herramienta en base a las primeras, focalizada sobre sus dos pilares fundamentales: estructura de base de datos (5.4.3.1) y código web (5.4.3.2). Finalmente, evaluaremos el grado de éxito de la implementación en base a las pruebas de aceptación, diseñadas para las historias de usuario planificadas, y acompañadas de capturas reales del sistema de forma que podamos apreciar el resultado final obtenido de cara a su despliegue (5.4.4).

5.4.1 Tareas de programación

Para mostrar el desglose de una historia de usuario en sus diferentes tareas de programación hemos seleccionado la historia de usuario: “Historia 4: Asociación de incidencia a contenido”, perteneciente a la segunda iteración de la primera liberación.

Cada tarea se representará mediante un identificador compuesto por el número de historia al que se hace referencia seguido del número de tarea para dicha historia, una breve descripción del objetivo de la

misma, y un conjunto de anotaciones sobre las que el programador se apoyará para realizar su implementación:

- **Tarea 4.1: Vías de asociación de incidencia.**

- **Descripción:** Tratamiento de las diferentes vías de asociación de incidencia posibles.
- **Anotaciones:**
 - 1) Únicamente disponible cuando el origen seleccionado es una agrupación lógica de ventanas. Definir en base de datos mediante un *flag* {0, 1} que indique si un determinado origen de incidencia constituye una agrupación lógica de ventanas (1) o no (0).
 - 2) La posibilidad de seleccionar una vía de asociación para una incidencia depende de la selección de una determinada acción reparadora. Si no existe una acción reparadora seleccionada no se posibilitará la selección de vías de asociación; en caso contrario, se obtendrán y representarán las vías posibles mediante un fichero PHP con conexión a BD parametrizado con el valor de la acción reparadora seleccionada. Este comportamiento se soportará mediante un evento JavaScript *onchange* en el combo de la acción reparadora, que bien capará la selección de la vía de asociación, o que lanzará la ejecución del fichero *PHP* mediante un objeto XMLHttpRequest asíncrono.
 - 3) La selección de las vías de asociación se modelará mediante una etiqueta HTML *select*. Las opciones posibles para la etiqueta se mantendrán en base de datos, mediante una relación entre acciones reparadoras y vías de asociación.

- **Tarea 4.2: Panel de asociación - Búsqueda de contenido.**

- **Descripción:** Tratamiento de los distintos mecanismos de búsqueda de contenido respecto al origen de incidencia posibles.
- **Anotaciones:**
 - 1) En caso de seleccionar la vía de asociación a contenido concreto, se nos presentarán diversas opciones de búsqueda en función de la acción reparadora seleccionada. Será necesario mantener en BD una relación entre las acciones reparadoras y las opciones de búsqueda permitidas.
 - 2) Únicamente se permitirá un tipo de búsqueda al mismo tiempo, por lo que la selección de la opción se modelará con *input* grupal de tipo radio. Un evento JavaScript *onclick* en cada una de las entradas lanzará una petición XMLHttpRequest al fichero PHP correspondiente en base al valor de la opción marcada.
 - 3) En caso de una selección de búsqueda por pedido, el fichero PHP con conexión a BD generará dinámicamente un listado de elementos seleccionables de tipo pedido en base al origen de incidencia especificado. Se detallarán para cada elemento el identificador de pedido y número de unidades contenidas en el mismo.
 - 4) En caso de una selección de búsqueda por modelo, el fichero PHP con conexión a BD generará dinámicamente un listado de elementos seleccionables de tipo modelo en base al origen de

incidencia especificado. Se detallarán para cada modelo el identificador de pedido, el identificador de modelo, y número de unidades contenidas en el mismo.

- 5) En caso de una selección de búsqueda por instancia, se mostrará un *input* de tipo texto, donde el usuario deberá introducir el código de barras asociado a la instancia deseada. Para cada evento JavaScript *onkeyup* ocurrido sobre este elemento, lanzaremos una nueva petición sobre un fichero PHP parametrizado con el contenido del campo de texto. En base a dicho código de barras, se comprobará la existencia en el sistema de una instancia asociada y vínculo de la misma con el origen de incidencia. Esta validación se realizará mediante una función almacenada SQL. En caso de una comprobación incorrecta, el PHP en cuestión retornará indicaciones para el usuario; en caso de una comprobación correcta, el PHP generará un elemento seleccionable de tipo instancia detallando la información presente en la etiqueta física asociada al código de barras especificado por el usuario.

- **Tarea 4.3: Panel de asociación - Contenido afectado.**

- **Descripción:** Gestión del listado de elementos afectados por la incidencia.

- **Anotaciones:**

- 1) En caso de seleccionar la vía de asociación a contenido concreto, se habilitará la formación de un listado de elementos afectados por la incidencia. Cada uno de estos elementos se modelará con un objeto de la clase PHP “ElementoIncidencia”, almacenados sobre un *array* contenido en una variable de sesión PHP. Esta clase deberá contener de igual forma un método capaz de transformar un elemento de tipo pedido o modelo en un elemento de tipo instancia si el contenido de estos se reduce a una única ventana. En cada ejecución, el fichero PHP encargado de la generación dinámica de este listado realizará una lectura de la variable de sesión y representará cada objeto como una entrada diferenciada en el listado.
- 2) Para traspasar un elemento obtenido en la búsqueda de contenido al listado de elementos afectados por la incidencia, se hará uso de un evento JavaScript *onclick* sobre cada uno, que lanzará asincrónicamente un PHP encargado de conformar un nuevo objeto “ElementoIncidencia” en base a la información del elemento seleccionado y añadirlo a la variable de sesión. Una vez obtenida la respuesta de esta ejecución se programará consecutivamente un refresco sobre el listado de elementos afectados, de forma que se visualicen los cambios aplicados. Para eliminar un registro del listado de elementos de incidencia se utilizará exactamente el mismo mecanismo. Para que el fichero PHP anteriormente mencionado distinga entre una operación de añadido o eliminado sobre la variable de sesión se le especificará mediante un elemento QueryString un valor operacional distinto.

5.4.2 Pruebas Unitarias

Como hemos expuesto en la planificación (5.3), las pruebas de unidad surgen a partir de las tareas de programación. Cada tarea de programación puede abarcar una o mas pruebas unitarias, que verifican la implementación detallada en dicha tarea. Siguiendo la misma lógica que para las tareas, identificaremos cada prueba de unidad con el identificador de tarea al que hace referencia seguido del número de prueba

para la misma, donde esta se conformará por una breve descripción, una entrada y una salida esperada. La cumplimentación de esta salida indicará el éxito o fracaso de la prueba. A continuación, exponemos algunos ejemplos de pruebas unitarias para la tarea previamente detallada en la sección 5.4.1: "Tarea 4.2: Panel de asociación - Búsqueda de contenido."

- **Prueba de unidad 4.2.1: Búsqueda por instancia - Código de barras no numérico.**
 - **Entrada:** El usuario introduce un código de barras con caracteres no numéricos.
 - **Salida Esperada:** El sistema muestra un mensaje de aviso, donde indica que el código de barras únicamente puede estar compuesto por caracteres numéricos.

- **Prueba de unidad 4.2.2: Búsqueda por instancia - Código de barras incompleto.**
 - **Entrada:** El usuario introduce un código de barras incompleto.
 - **Salida Esperada:** El sistema muestra una indicación, donde indica al usuario el número de dígitos faltantes para llegar a los 14 que componen el código de barras.

- **Prueba de unidad 4.2.3: Búsqueda por instancia - Código de barras completo inexistente.**
 - **Entrada:** El usuario introduce un código de barras completo, pero inexistente en el sistema.
 - **Salida Esperada:** El sistema muestra un mensaje de información, donde indica al usuario que el código introducido no está asociado a ninguna instancia presente en el sistema.

- **Prueba de unidad 4.2.4: Búsqueda por instancia - Código de barras existente independiente del origen de incidencia.**
 - **Entrada:** El usuario introduce un código de barras completo, existente en el sistema, sin relación con el origen.
 - **Salida Esperada:** El sistema muestra un mensaje de error, donde indica al usuario que el código introducido no identifica una instancia contenida en el origen de incidencia especificado.

- **Prueba de unidad 4.2.5: Búsqueda por instancia - Código de barras existente contenido en el origen de incidencia.**
 - **Entrada:** El usuario introduce un código de barras completo, existente en el sistema, y que identifica una instancia concreta contenida en el origen de incidencia dado.
 - **Salida Esperada:** El sistema muestra la información asociada a la instancia presente en la etiqueta física, además de permitir la selección del elemento para conformar la lista de contenido afectado por la incidencia.

5.4.3 Implementación

En esta sección nos centraremos en mostrar las estructuras sobre las que se ha cimentado la herramienta, tanto desde el desarrollo web como desde el trabajo de base de datos. Para el primero describiremos la arquitectura *software* llevada a la práctica en base al entorno de trabajo utilizado (5.4.3.2), mientras que para el segundo (5.4.3.1) detallaremos el esquema relacional de base de datos diseñado para soportar la necesidad de información del sistema.

5.4.3.1 Base de datos (BD)

En esta sección detallamos el diagrama de base de datos relacional, diseñado sobre el sistema de gestión de base de datos Microsoft SQLServer, que da soporte a la herramienta. Además de las tablas creadas específicamente para el desarrollo del proyecto, debido a la implicación de conceptos ya existentes, ha sido necesario realizar un trabajo de vinculación entre las primeras y estructuras presentes de forma previa en el sistema. En el diagrama de base de datos diferenciamos estas tablas y sus correspondientes relaciones mediante un sombreado azul. El esquema de base de datos desarrollado, correspondiente al despliegue de la tercera liberación y por tanto con forma final, contaría con la siguiente estructura:

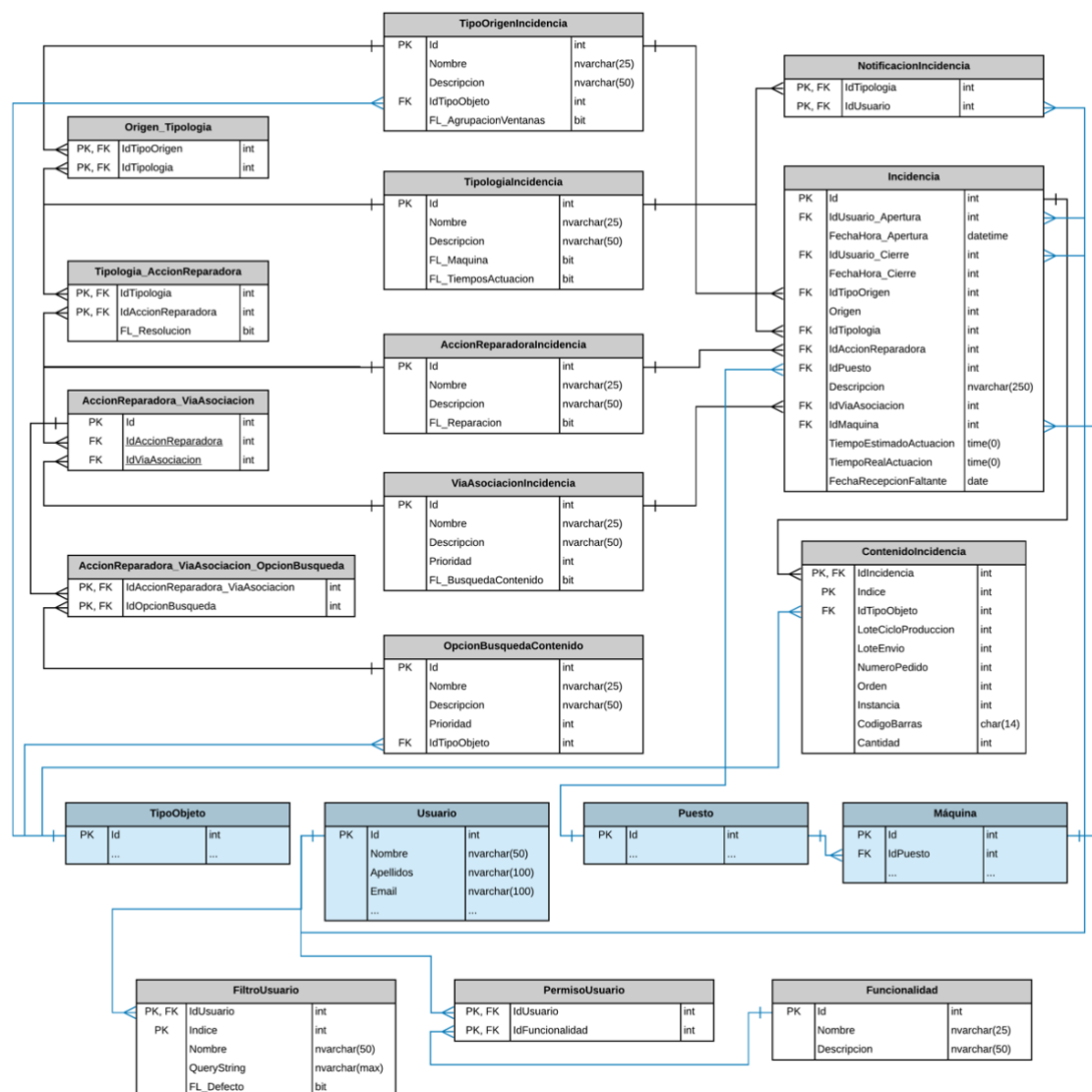


Ilustración 5.5: Diagrama BD correspondiente a la tercera liberación

A continuación, se proporciona una breve explicación de cada tabla presente en el diagrama, así como su cometido:

- **TipoObjeto:** Estructura ya existente. Identifica las distintas naturalezas de objeto presentes en el sistema. Para nuestro desarrollo nos interesarán: lote-ciclo de producción, lote de envío, pedido, modelo, instancia, cuadro (marco y hoja) y pieza (marco, hoja y travesaño).
- **Usuario:** Estructura ya existente. Identifica los usuarios presentes en el sistema. Podemos obtener información relevante como nombre, apellidos, o correo electrónico asociado.
- **Puesto:** Estructura ya existente. Identifica los puestos de trabajo distribuidos a lo largo de la fábrica.
- **Maquina:** Estructura ya existente. Identifica las distintas máquinas de producción distribuidas a lo largo de la fábrica. Cada máquina se asocia al puesto en el que esta se encuentra.
- **TipoOrigenIncidencia:** Mantiene los distintos tipos de origen desde los que puede introducirse una incidencia. Identifica el tipo de objeto al que el origen en cuestión hace referencia, así como un indicador sobre si el mismo constituye una organización lógica de ventanas.
- **TipologiaIncidencia:** Mantiene las distintas tipologías de incidencia existentes en el sistema. Además, indica si una determinada tipología implica una especificación de máquina y/o una especificación de tiempos de actuación.
- **AccionReparadoraIncidencia:** Mantiene las distintas acciones reparadoras de incidencia existentes en el sistema. Además, indica si una determinada acción reparadora conlleva reparaciones.
- **ViaAsociacionIncidencia:** Mantiene las distintas vías de asociación de incidencia a contenido existentes en el sistema. Además, establece su prioridad, que definirá la selección por defecto en caso de varias vías de asociación disponibles. También indica si la vía de asociación corresponde a una selección de contenido concreto.
- **OpcionBusquedaContenido:** Mantiene las distintas opciones de búsqueda de contenido respecto al origen existentes en el sistema. Además, establece su prioridad, que definirá la selección por defecto en caso de varias opciones de búsqueda de contenido disponibles. También identifica el tipo de objeto al que la opción de búsqueda en cuestión hace referencia.
- **Origen_Tipologia:** Gestiona las relaciones entre el tipo de origen y las tipologías de incidencia. Este diseño posibilita que cada tipo de origen pueda involucrar distintas tipologías.
- **Tipologia_AccionReparadora:** Gestiona las relaciones entre la tipología y las acciones reparadoras de incidencia. Este diseño posibilita que cada tipología pueda involucrar distintas acciones reparadoras. Además, debido a que se puede requerir una resolución de incidencia tanto en base a la tipología como a la acción reparadora, hemos decidido incluir dicho indicador en esta tabla de relación.
- **AccionReparadora_ViaAsociacion:** Gestiona las relaciones entre la acción reparadora y las vías de asociación de incidencia. Este diseño posibilita que cada acción reparadora pueda involucrar distintas vías de asociación. Representamos la dupla única [acción reparadora, vía asociación] mediante un identificador único de relación.

- **AccionReparadora_ViaAsociacion_OpcionBusqueda:** Gestiona las relaciones entre la dupla [acción reparadora, vía asociación] y las opciones de búsqueda de contenido. Este diseño posibilita que cada composición [acción reparadora, vía asociación] pueda involucrar distintas opciones de búsqueda.
- **NotificacionIncidencia:** Gestiona las relaciones entre la tipología de incidencia y los usuarios que deben ser notificados mediante correo electrónico. Este diseño posibilita que cada tipología de incidencia pueda involucrar notificaciones a distintos usuarios.
- **Incidencia:** Almacena la información descriptiva de una incidencia registrada en el sistema: su composición y características. Se trata de la estructura de datos primordial.
- **ContenidoIncidencia:** Asocia una determinada incidencia al contenido respecto al origen afectado por la misma; pudiendo este contenido responder a objetos de distintos tipos o naturalezas.
- **Funcionalidad:** Mantiene las distintas funcionalidades susceptibles de restricción a lo largo del sistema.
- **PermisoUsuario:** Detalla las distintas funcionalidades para las que un determinado usuario tiene un acceso garantizado.
- **FiltroUsuario:** Mantiene los distintos filtros configurados por el usuario para la búsqueda de incidencias mediante el almacenamiento de la cadena de consulta resultante. Además, indica si un determinado filtro debe cargarse por defecto.

5.4.3.2 WEB

Un entorno de trabajo o *framework* puede definirse como una estructura de artefactos *software* que colaboran bajo un conjunto estandarizado de conceptos, criterios y prácticas, con el fin de facilitar una arquitectura de reutilización que conforme una base para la organización y desarrollo de aplicaciones relacionadas.

Para el desarrollo de esta aplicación se ha empleado un entorno de trabajo definido por la propia empresa AGI10. Normalmente, los distintos proyectos que se van desarrollando son responsabilidad de un único integrante del departamento, por lo que las decisiones en cuanto a la implementación del sistema, así como las prácticas llevadas a cabo tomaban una dependencia unilateral. Esta situación podría presentar inconvenientes cuando un integrante no involucrado en el desarrollo de una herramienta se vea a posteriori obligado a realizar mantenimiento sobre la misma. De esta forma AGI10 ha diseñado un entorno de trabajo genérico sobre el cual deben construirse todas las aplicaciones de similares características; en nuestro caso, un sistema web de información basado en transacciones de base de datos. A continuación, se muestra y define la estructura de ficheros acordada para este tipo de sistemas:



Ilustración 5.6: Estructura de ficheros WEB

- El fichero “index.php” representa la página principal de la aplicación.
- La carpeta “pages” contiene un fichero de cabecera para cada sección o módulo de la aplicación. Cada fichero se denomina como el módulo al que hace referencia. A su vez se habilita una nueva carpeta para cada uno de estos módulos con idéntica nomenclatura, la cual contiene los ficheros PHP que componen sus funcionalidades. Estos ficheros son los encargados de comunicarse con la base de datos y generar dinámicamente código HTML. Cada fichero se denomina de forma acorde a la funcionalidad a la que hace referencia.

- La carpeta “scripts” se subdivide en las carpetas “php” y “javascript”. Cada una de ellas presenta un fichero para utilidades comunes. En la carpeta “javascript” contamos con un fichero para cada módulo, encargado de gestionar la interacción con el usuario. Además, se mantienen los ficheros referentes a las distintas clases necesarias para ambas tecnologías.
- La carpeta “style” se subdivide en las carpetas “css” e “img”. La primera identifica las hojas de estilo aplicadas en la herramienta. Se define una hoja de estilo por cada módulo del sistema, además de una hoja de carácter general para aquellos elementos presentes transversalmente en la interfaz. En la subcarpeta “img” se almacenan todos los elementos gráficos empleados.
- La carpeta “plugins” almacena los distintos complementos adicionales utilizados en la aplicación.

En base a la estructura recién definida y a modo de ejemplo, mostramos un esquema de comunicación asíncrona entre los distintos componentes para conformar una respuesta ante una interacción del usuario:

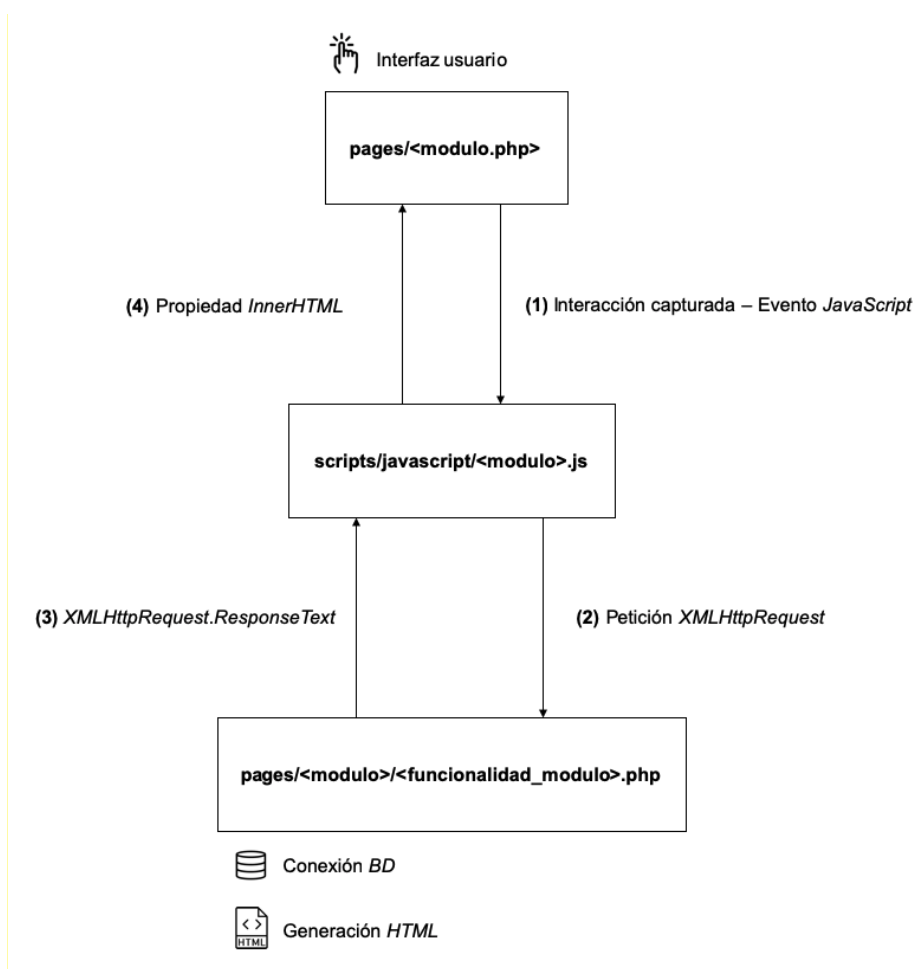


Ilustración 5.7: Esquema comunicación asíncrona en nuestro entorno de trabajo

1. Partimos de la interfaz de usuario, conformada por el código HTML del módulo en el que nos encontremos actualmente: “pages/<módulo>.php”. El usuario interactúa con la aplicación.

2. Esta interacción es capturada por el evento JavaScript correspondiente. El evento vincula la ejecución de una función presente en el fichero “scripts/javascript/<modulo>.js”.
3. Por medio de esta función lanzamos un objeto XMLHttpRequest sobre el fichero “pages/<modulo>/<funcionalidad_modulo>.php” asociado.
4. Estos ficheros, parametrizados y con conexión a BD, generan dinámicamente el código HTML de respuesta.
5. La función JavaScript recibe el código proporcionado por el servidor, y lo retorna al módulo en cuestión mediante la propiedad *innerHTML* del elemento apropiado.
6. La interfaz de usuario queda preparada para una nueva interacción, retornando el proceso al primer paso.

5.4.4 Pruebas de aceptación

A diferencia de las pruebas de unidad, diseñadas para corroborar las tareas de programación, las pruebas de aceptación surgen directamente de las historias de usuario. Una historia de usuario puede abarcar una o mas pruebas de aceptación, en función de las que sean necesarias para verificar la correcta y completa funcionalidad definida en cada una de las historias. Las pruebas de aceptación requieren de una gran implicación del cliente, encargado de ayudar en su desarrollo y de proporcionar datos reales sobre los que se realizarán las mismas, con el fin de comprobar la correcta implementación de estas en un entorno cercano al de producción.

La definición de estas pruebas de aceptación seguirá un patrón similar al de las pruebas unitarias. Identificaremos cada prueba de aceptación con el número de historia al que hace referencia seguido del número de prueba para la misma, donde esta se conformará por una breve descripción, un flujo de entrada y un flujo de salida esperada. La cumplimentación de este flujo de salida indicará el éxito o fracaso de la prueba. A continuación, exponemos algunos ejemplos de pruebas aceptación, junto capturas reales del sistema resultante para un escenario de registro, gestión y resolución de incidencia.

- **Prueba de aceptación 1.1: Selección origen de incidencia - Lote/Ciclo de producción.**

- ***Flujo Entrada.***

- 1) El usuario accede a la utilidad de alta de incidencias internas.
- 2) El usuario selecciona el origen de incidencia “Lote / Ciclo de Producción”.
- 3) El usuario introduce el lote-ciclo de producción “123401”.
- 4) El usuario introduce el lote-ciclo de producción “964701”.

- ***Flujo Salida Esperado.***

- 1) El sistema solicita al usuario la selección de un tipo de origen de incidencia entre los distintos posibles.
- 2) El sistema habilita la introducción de un identificador numérico para el origen, al tratarse de una agrupación lógica de ventanas.
- 3) El sistema muestra un mensaje de advertencia, en el que indica que el lote-ciclo de producción especificado no existe en el sistema.

- 4) El sistema habilita el formulario de introducción de incidencia, al comprobar que el lote-ciclo de producción introducido existe en el sistema.

Lote / Ciclo de Producción ▼	123401 🔍
------------------------------	-----------

¡Cuidado! No existe ningún Lote 1234 - Ciclo 1 de Producción

Revise la información introducida

Ilustración 5.8: Prueba de aceptación 1.1 – Origen inexistente

- **Prueba de aceptación 2.1: Composición de incidencia - Lote/Ciclo de producción.**

- **Flujo Entrada.**

- 1) <Partimos de prueba de aceptación 1.1, flujo de entrada 4>
- 2) El usuario selecciona la tipología de incidencia “Daños”.
- 3) El usuario selecciona el puesto de detección, y pulsa “Registrar Incidencia”.
- 4) El usuario selecciona la tipología “Falta de Material” y aporta una descripción de incidencia.
- 5) El usuario selecciona la acción reparadora “Informar Responsables”.

- **Flujo Salida Esperado.**

- 1) El sistema muestra el formulario de introducción de incidencia, compuesto por los campos: tipología de incidencia, acción reparadora, puesto de detección y descripción de la incidencia. Además, informa tanto del tipo de origen (lote de producción) y origen concreto (964701) sobre el que se acometerá la introducción, como de la fecha de alta y usuario responsable. Igualmente, el sistema filtra las tipologías disponibles de acuerdo al tipo de origen seleccionado.
- 2) El sistema filtra las acciones reparadoras disponibles de acuerdo a la tipología seleccionada.
- 3) El sistema muestra un mensaje de advertencia, en el que indica al usuario que debe especificar todos los campos señalados como requeridos.
- 4) El sistema filtra las acciones reparadoras disponibles de acuerdo a la tipología seleccionada. En este caso:
 - Informar responsables.
- 5) El sistema habilita la selección de vías de asociación de incidencia, al tratarse de una agrupación lógica de ventanas.

- **Prueba de aceptación 4.1: Asociación de incidencia - Contenido concreto de diferentes naturalezas respecto a Lote/Ciclo de producción.**

- ***Flujo Entrada.***

- 1) <Partimos de prueba de aceptación 2.1, flujo de entrada 5>
- 2) El usuario selecciona la vía de asociación “Asociar a Pedidos o Ventanas”.
- 3) El usuario marca como afectado el pedido “920686”.
- 4) El usuario vuelve a marcar como afectado el pedido “920686”.
- 5) El usuario selecciona la opción de búsqueda de contenido “Modelo”.
- 6) El usuario selecciona el pedido “921728”.
- 7) El usuario marca como afectado el modelo “V1A” del pedido “921728”.
- 8) El usuario marca como afectado el modelo “V1B” del pedido “921728”.
- 9) El usuario selecciona la opción de búsqueda de contenido “Instancia”.
- 10) El usuario lee o introduce el código de barras “96470046010011”.
- 11) El usuario marca como afectada la instancia resultante (pedido “922079”, modelo “V1”, instancia 1).
- 12) El usuario desmarca como afectado el modelo “V1B” del pedido “921728”.
- 13) El usuario pulsa “Registrar incidencia”.

- ***Flujo Salida Esperado.***

- 1) El sistema muestra las vías de asociación disponibles. En este caso, debido a la acción reparadora seleccionada:
 - Asociar a todo el lote-ciclo de producción (origen completo). Por defecto, en base a la prioridad definida.
 - Asociar a pedidos o ventanas (contenido concreto).
- 2) El sistema muestra por defecto las opciones de búsqueda de contenido disponibles. En este caso, debido a la acción reparadora y la vía de asociación seleccionada:
 - Pedido. Al estar marcada por defecto en base a la prioridad definida, el sistema muestra directamente el listado de los diferentes pedidos contenidos en el origen de incidencia junto a las unidades que estos implican.
 - Modelo.
 - Instancia.
- 3) El sistema traspa el pedido seleccionado al listado de elementos afectados por la incidencia.
- 4) El sistema comprueba que el elemento ya está incluido en el listado de elementos afectados por la incidencia, por lo que no hace nada.

- 5) El sistema muestra el listado de los diferentes pedidos contenidos en el origen de incidencia, e indica al usuario que debe seleccionar uno de ellos para ver el listado de modelos contenidos en el mismo.
- 6) El sistema muestra el listado de los diferentes modelos contenidos en el pedido seleccionado junto a las unidades que estos implican.
- 7) El sistema traspasa el modelo seleccionado al listado de elementos afectados por la incidencia.
- 8) El sistema detecta que el modelo seleccionado contiene una única unidad, por lo que hace una conversión del elemento a la naturaleza instancia y lo traspasa al listado de elementos afectados por la incidencia.
- 9) El sistema habilita la introducción de un código de barras, e indica al usuario que para identificar una instancia puede introducir los dígitos correspondientes a cualquier etiqueta presente en la misma.
- 10) El sistema comprueba la validez del código introducido y su asociación con el origen de incidencia, y obtiene la instancia asociada junto con la información presente en la etiqueta física del producto.
- 11) El sistema traspasa la instancia seleccionada al listado de elementos afectados por la incidencia.
- 12) El sistema elimina el modelo seleccionado del listado de elementos afectados por la incidencia.
- 13) El sistema muestra un mensaje de éxito, en el que indica al usuario que la incidencia ha quedado registrada correctamente, además de especificar el identificador de incidencia asociado a esta.

Nueva Incidencia • Lote 9647 (01)
Jairo Diego Cuesta
4 Septiembre 2019

Falta de Material

Informar a Responsables

Acrystalado Línea Medida

Referencia XYZ (Prueba Sistemas)

Asociación Incidencia

Asociar a Pedidos o Ventanas

☐ Pedido

☐ Modelo

☒ Instancia

Para añadir una nueva instancia, introduzca los 14 dígitos del código de barras presente en cualquier etiqueta de la ventana

>>>

Pedido 922079, V1 • INSTANCIA 1	×
Pedido 921728, V1B • INSTANCIA 1	×
Pedido 921728, V1A • 2 uds.	×
Pedido 920686 • 4 uds.	×

Registrar Incidencia

* Campos Requeridos

Ilustración 5.9: Prueba de aceptación 4.1 – Formulario de introducción

- 41 -

- **Prueba de aceptación 5.1: Notificación de apertura vía correo electrónico - Incidencia de falta de material asociada a elementos de distintas naturalezas respecto a un Lote/Ciclo de producción.**

- **Flujo Entrada.**

1) <Partimos de prueba de aceptación 4.1, flujo de entrada 13>

- **Flujo Salida Esperado.**

1) El sistema, al tratarse de una tipología de incidencia con notificación activa, envía un correo electrónico a todos los usuarios definidos para dicha tipología con la información completa de la incidencia recién dada de alta.



Ilustración 5.10: Prueba de aceptación 5.1 – Correo de notificación

- **Prueba de aceptación 6.1: Listado de incidencias registradas - Acceso.**

- **Flujo Entrada.**

1) El usuario accede al módulo de búsqueda de incidencias.

- **Flujo Salida Esperado.**

1) El sistema muestra al usuario un listado con las distintas incidencias registradas en el sistema, donde cada entrada del mismo representa una incidencia diferente. Cada entrada mantiene la información de apertura (identificador de incidencia, fecha y hora, usuario y puesto), tipología de incidencia, acción reparadora e información de afección (origen, contenido, descripción y máquina si se aplicase).

Búsqueda Incidencia

Incidentes Internas | Búsqueda Incidencia

Incidentes Internas

Nueva Incidencia

Búsqueda Incidencia

FILTRADO

Ningún filtro seleccionado

ESTADO

Todos

ID / DESCRIPCIÓN

ID o texto a buscar

FECHA APERTURA

Desde

Hasta

PUESTO DETECCIÓN

Todos

TIPOLOGÍA

Todas

ACC. REPARADORA

Todas

IMPLICACIONES ESPECIALES

Recepción de faltante

Reparaciones

Máquina

Guardar Filtro

Limpiar Filtro

LISTADO DE INCIDENCIAS

APERTURA	TIPOLOGÍA	ACC. REPARADORA	AFECTADO	ESTADO
# 26000 4 Septiembre 2019 • 17:08 Javier Bedia Viadero Especiales PVC 2	Falta de Material	Informar a Responsables	3 Ud/s. del LP 9847 (01) "faltan tiradores"	En Curso Sin recepción estimada
# 25999 4 Septiembre 2019 • 16:54 Nieves Carton Lopez Corte de Básicos	Falta de Material	Informar a Responsables	1 Ud/s. del LP 9917 (01) "FALTA GUIA REF-43500AG LA QUE HAY ESTA RAYADA... Y FALTA GUIA DE 85 EN AG"	En Curso Sin recepción estimada
# 25998 4 Septiembre 2019 • 16:47 Javier Rozas Gutierrez Gestion Vidrio	Falta o Incidencia de Vidrio	Informar a Responsables	2 Ud/s. del LP 9985 (01) "vidrios por pedido sin pedir"	En Curso Sin recepción estimada
# 25997 4 Septiembre 2019 • 16:42 Javier Rozas Gutierrez Gestion Vidrio	Falta o Incidencia de Vidrio	Informar a Responsables	2 Ud/s. del LP 9985 (01) "entrega estimada de vidrios 05/09 con tvitec y jucar"	En Curso Sin recepción estimada
# 25996 4 Septiembre 2019 • 16:37 Ismael Inero Hierro Herraje de Marcos	Operación mal realizada por Operario	Repetir Marco	1 Ud/s. del LP 9663 (01) "mal soldado pA LOS CAMBIADOS"	Resuelta 4 Septiembre 2019 • 17:50 Jesus Fuentes Hernaiz
# 25995 4 Septiembre 2019 • 16:36 Nieves Carton Lopez Corte de Básicos	Falta de Material	Informar a Responsables	15 Ud/s. del LP 9910 (01) "falta ensanchador de 80 en blanco sapelly...va SUELTO!!"	En Curso Sin recepción estimada
# 25994 4 Septiembre 2019 • 15:18 Adrian De La Rosa Fernández	Daños	Repetir Marco	1 Ud/s. del LP 9663 (01) "bollo levantado"	Resuelta 4 Septiembre 2019 • 16:26 Jesus Fuentes Hernaiz

Mostrando resultados 1-10 de 100

1 2 3 4 5 »

Mostrar de 10 en 10

Incidencia "En Curso". Demanda una resolución, pero aún no cuenta con fecha de cierre.

Incidencia "Resuelta". Demanda una resolución, y ya cuenta con fecha de cierre.

Incidencia "Notificada". No demanda ningún tipo de gestión. Meramente informativa.

Ilustración 5.11: Prueba de aceptación 6.1 – Listado de incidencias (completo)

- Prueba de aceptación 9.1: Filtrado de incidencias - Búsqueda por ID.**
 - Flujo Entrada.**
 - El usuario introduce el identificador de incidencia único “25981”.
 - Flujo Salida Esperado.**
 - El sistema actualiza el listado de incidencias, mostrando únicamente la entrada asociada al identificador de incidencia especificado.

LISTADO DE INCIDENCIAS				
APERTURA	TIPOLOGÍA	ACC. REPARADORA	AFECTADO	ESTADO
# 25981 4 Septiembre 2019 • 11:26 Jairo Diego Cuesta Acristalado Línea Medida	Falta de Material	Informar a Responsables	7 Ud/s. del LP 9647 (01) "Referencia XYZ (Prueba Sistemas)"	En Curso Sin recepción estimada

Ilustración 5.12: Prueba de aceptación 9.1 – Listado de incidencias (detalle)

- 43 -

- **Prueba de aceptación 6.2: Listado de incidencias registradas - Visualización contenido afectado.**

- **Flujo Entrada.**

- 1) El usuario selecciona la opción de visualización detallada del contenido afectado respecto al origen, para la incidencia “25981”.
- 2) El usuario cierra la ventana emergente.

- **Flujo Salida Esperado.**

- 1) El sistema muestra una ventana emergente, donde detalla cada elemento afectado por la incidencia, identificando la naturaleza del mismo y su definición. En este caso:
 - Pedido 920686.
 - Pedido 921728 - Modelo V1A.
 - Pedido 922079 - Modelo V1 - Instancia 1.
- 2) El sistema vuelve a la visualización del listado de incidencias.

- **Prueba de aceptación 12.1: Gestión de incidencia - Faltante de material.**

- **Flujo Entrada.**

- 1) El usuario despliega el menú de acciones para la entrada referente a la incidencia “25981”.
- 2) El usuario selecciona la edición de la fecha de recepción estimada.
- 3) El usuario establece la fecha estimada de recepción para el día “6 de septiembre de 2019” y aplica los cambios.

- **Flujo Salida Esperado.**

- 1) El sistema presenta al usuario las distintas acciones disponibles. En este caso:
 - Editar fecha de recepción estimada
 - Cerrar incidencia
- 2) El sistema muestra una ventana emergente, donde propone al usuario la selección de una fecha de recepción estimada.
- 3) El sistema vuelve a la visualización del listado de incidencias e informa al usuario mediante un mensaje de éxito de que la fecha estimada de recepción se ha modificado correctamente.

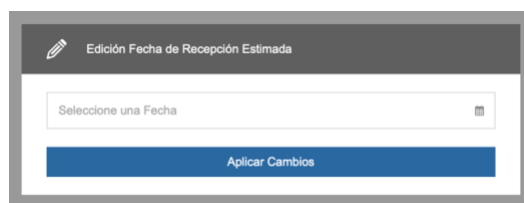


Ilustración 5.13: Prueba de aceptación 12.1 – Edición fecha estimada de recepción

- **Prueba de aceptación 12.2: Gestión de incidencia - Cerrado de incidencia.**
 - **Flujo Entrada.**
 - 1) El usuario despliega el menú de acciones para la entrada referente a la incidencia “25981”. El usuario selecciona el cierre de incidencia.
 - **Flujo Salida Esperado.**
 - 1) El sistema presenta al usuario las distintas acciones disponibles. En este caso:
 - Editar fecha de recepción estimada
 - Cerrar incidencia
 - 2) El sistema vuelve a la visualización del listado de incidencias e informa al usuario mediante un mensaje de éxito de que esta se ha cerrado correctamente.
- **Prueba de aceptación 14.1: Estado de incidencia - Incidencia resuelta.**
 - **Flujo Entrada.**
 - 1) <Partimos de prueba de aceptación 12.2, flujo de entrada 2>
 - **Flujo Salida Esperado.**
 - 1) El sistema actualiza el estado de la entrada del listado referente a la incidencia, donde indica que esta se encuentra resuelta, además de el usuario y fecha de cierre.

● **Resuelta**
 ⌚ 4 Septiembre 2019 • 18:15
 👤 Jairo Diego Cuesta

Ilustración 5.14: Prueba de aceptación 14.1 – Columna “estado”

6 Trabajo futuro

En este capítulo (6) detallaremos algunos de los desarrollos no abarcados a esta altura del proyecto, pero que podrían ser incluidos en futuras liberaciones mediante nuevas historias de usuario con el objetivo de añadir valor a la herramienta, así como de completar su funcionalidad.

Dividiremos estos trabajos en dos vertientes. Por un lado, tenemos un conjunto de necesidades que ya han sido atisbadas y planteadas informalmente por el cliente de cara a nuevos despliegues (6.1), y por el otro lado contamos con una serie de funcionalidades no discutidas por el momento, pero que desde el equipo de desarrollo creemos que podrían aportar posibilidades de gran interés (6.2).

6.1 Garantizados

1. **Edición, borrado y reapertura de incidencias:** Mediante estas funcionalidades un usuario con privilegios sería capaz de editar, borrar o reabrir una incidencia. En ciertas ocasiones se puede cometer un error en la especificación de una incidencia, testar un determinado flujo de acción mediante registros de prueba, o descubrir que una incidencia marcada como resuelta no debía ser considerada como tal. En todas estas situaciones los controles expuestos conformarían una solución para las mismas. Actualmente, este tipo de actuaciones son llevadas a cabo por el departamento de sistemas mediante una manipulación directa de los datos, pero sería conveniente que pudieran ser gestionadas autónomamente por los usuarios adecuados.
2. **Evidencias:** Posibilitar que un usuario pueda adjuntar documentos gráficos o relevantes en la apertura de una determinada incidencia, con el fin de proporcionar una justificación para la misma o facilitar su resolución.
3. **Seguimiento de incidencia:** Consistiría en habilitar un registro histórico, donde usuarios relevantes tuvieran la posibilidad de añadir comentarios respecto a una incidencia. Este desarrollo aportaría un gran valor de cara al conocimiento exacto del correspondiente proceso de resolución.
4. **Acciones correctivas:** Una acción correctiva, tal como en AGI10 se entiende, podría definirse como un conjunto de trabajos a realizar con el fin de poner una solución definitiva a una incidencia recurrente, surgida no a partir de un hecho puntual, sino a partir de un problema de base. Consistiría en implementar una nueva utilidad que permitiese añadir y asociar acciones correctivas a incidencias existentes.

6.2 A debate

5. **Configuración de la herramienta:** Orientado principalmente a la configuración de información maestra para usuarios con privilegios: tipologías de incidencias, acciones reparadoras, tablas de relación, etc. Actualmente llevado a cabo por el departamento de sistemas mediante manipulación directa de los datos.
6. **“Business Intelligence” o inteligencia empresarial:** Elaboración de informes con *software* de inteligencia empresarial, como Microsoft Power BI, publicados directamente en nuestra propia herramienta. Permitiría que usuarios con privilegios tuvieran acceso a una visualización gráfica en tiempo real de las situaciones de incidencia en fábrica.
7. **Exportación de incidencias:** Posibilidad de que un usuario pueda exportar a un archivo Excel un listado de incidencias acorde a un determinado filtro. Permitiría que cada usuario pudiese realizar un análisis privado de los datos de acuerdo a la necesidad de su propio departamento.
8. **Acciones correctivas + IA:** Como hemos expuesto anteriormente, una acción correctiva propone acabar con una situación de incidencia recurrente. Podría analizarse dicha recurrencia con técnicas de inteligencia artificial, por ejemplo, NLP (procesamiento del lenguaje humano), que detectasen patrones en las incidencias registradas y propusiesen la creación de una nueva acción correctiva para incidencias de similares características o la asociación de la misma a una acción correctiva ya existente.

7 Conclusiones

Hemos decidido dotar a las conclusiones resultantes del proyecto de dos enfoques diferenciados. Primero, midiendo el grado de éxito del mismo puramente en base a la herramienta resultante (7.1), y a posteriori desde un punto de vista mas académico y personal (7.2).

7.1 Herramienta

Un gran indicador para medir el éxito del desarrollo sería la propia experiencia y sensación de satisfacción del usuario final. A muy alto nivel, podemos diferenciar entre un perfil de trabajador de línea y un perfil de responsable. El primero se ha encontrado con un sistema altamente usable, basado en una interfaz limpia e intuitiva, que le permite de una forma sencilla notificar una situación de incidencia con la versatilidad y profundidad necesarias para abarcar todas las circunstancias posibles. En cuanto a los intereses del segundo, cuenta con un sistema de acceso y lectura de datos igualmente intuitivo a la par que potente, proporcionándole capacidades de seguimiento, gestión y resolución de incidencias. Podemos comprobar cómo todas estas propiedades derivan de los objetivos planteados la sección 2.4 del presente informe, y en ambos casos resultan en una facilitación del trabajo y su flujo. En definitiva, se ha comprobado que la experiencia del usuario es altamente positiva, y para contrastarlo podemos analizar breve y objetivamente algunos aspectos acerca de la información de uso de la herramienta:

Desde el 25 de marzo de 2019, fecha en la que se despliega la primera liberación y con ella la recolección de incidencias, hasta el día 1 septiembre del mismo año (momento en el que se redactan estas líneas), el sistema cuenta con un total de **7895 incidencias** registradas, las cuales asocian un total de **8989 elementos** provenientes de diferentes naturalezas; donde la apertura de las mismas se distribuye a lo largo de **162 usuarios**. En base a la realización de una media aritmética pura y ciertamente generosa, donde tendríamos en cuenta el tiempo transcurrido desde el primer despliegue hasta el día actual, abarcando periodos de 24 horas y festivos, resulta en una cadencia de **registro de 29 minutos**.

Por otro lado, partiendo de la fecha de la tercera liberación, 20 de mayo de 2019, mediante la cual se habilitaba la gestión de incidencias, el sistema ha contabilizado un total de **4130 registros con implicación de resolución**, donde actualmente se cuenta con una gestión y correspondiente cierre del **98%** de las mismas.

Un indicador fiable para medir la optimización en la resolución de incidencias puede obtenerse a partir de aquellos registros que impliquen reparaciones, donde todas conllevan un proceso similar, a diferencia de las actuaciones en máquina (complejidades muy variables), o la recepción de faltante (disponibilidad del proveedor). En base a este tipo de incidencias, hemos podido concluir que el tiempo de resolución para las mismas, transcurrido desde el momento exacto de su apertura hasta su cierre, se ha visto **mejorado en un 50%**, pasando de un tiempo medio de resolución de **4 horas a un valor óptimo de 2 horas**.

7.2 Académico/Personal

Desde una perspectiva académica y personal, la experiencia ha sido completamente satisfactoria. A lo largo del desarrollo del proyecto, ha sido posible corroborar progresivamente la importancia y ventajas de haber realizado este trabajo de fin de grado en una empresa, en este caso, AGI10. Haber podido aplicar de forma práctica los conocimientos adquiridos a lo largo de la titulación sobre un entorno de producción real, ha aportado una serie de competencias extra, que conformarán una base y ayuda de cara a un futuro laboral.

Muestra de ello sería, por ejemplo, la puesta en marcha de una metodología ágil de desarrollo *software* para llevar a cabo el proyecto. En este caso, se ha podido comprobar como los preceptos presentados en la programación extrema no son meramente teóricos, sino que en una situación de desarrollo real son capaces de optimizar de forma sobresaliente el proceso de construcción de *software*.

De igual forma se ha podido verificar que, independientemente del área de negocio, objetivo o producto de la empresa en cuestión, la informática tiene un peso fundamental en la misma, y que precisamente el incremento de dicho peso es el que a día de hoy puede marcar la diferencia con sus principales competidores. En esta situación concreta, la elaboración de un sistema de recolección y gestión de incidencias ha repercutido directamente en la propia producción de ventanas. Como futuros graduados en ingeniería informática, el abanico de posibilidades que se nos presenta y del que disponemos actualmente es excepcional.

Por último, el desarrollo de este proyecto ha permitido certificar como el denominado “cliente”, y por tanto las personas que así lo conforman, emplean el producto pensado y diseñado con el objetivo de cubrir, de una forma u otra, parte de las necesidades diarias implicadas en el desempeño de su cometido, y, por tanto, facilitando el mismo. Al fin y al cabo, la tarea primordial de un ingeniero es la de dibujar una solución para un problema con un propósito concreto, siendo capaz de generar un efecto positivo. El hecho de observar como los diferentes compañeros, ya sea a lo largo de la cadena de montaje o en las propias oficinas, emplean y se aprovechan de la solución planteada para ellos, ha generado un sentimiento de privilegio y satisfacción con un valor personal idéntico al del resto de conclusiones sacadas sobre la realización de este trabajo.

8 Bibliografía

- [1] I. Sommerville, Ingeniería de Software, vol. 9, Pearson, 2011.
- [2] L. Apke, Understanding the Agile Manifesto, 2015.
- [3] K. Beck, Extreme Programming Explained: Embrace Change, vol. 2, 2005.
- [4] «Tecnologías WEB,» [En línea]. Available: <https://www.w3schools.com>.
- [5] «JavaScript,» [En línea]. Available: <https://www.javascript.com>.
- [6] «PHP,» [En línea]. Available: <https://www.php.net>.
- [7] Symfony, «Swift Mailer,» [En línea]. Available: <https://swiftmailer.symfony.com>.
- [8] «Microsoft SQL Server,» [En línea]. Available: <https://www.microsoft.com/es-es/sql-server>.
- [9] «Transact-SQL,» [En línea]. Available: <https://docs.microsoft.com/es-es/sql/t-sql/>.
- [10] «Microsoft Office,» [En línea]. Available: <https://www.office.com>. [Último acceso: Julio].
- [11] «Microsoft Power BI,» [En línea]. Available: <https://powerbi.microsoft.com/>.
- [12] S. Ambler, Agile Modelinng: Effective Practices for eXtreme Programming and the Unified Process, 2002.
- [13] M. Holcombe, Running an Agile Software Development Project, 2008.
- [14] «Extreme Programming,» [En línea]. Available: <http://www.extremeprogramming.org>.